NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

FLIGHT REGIME RECOGNITION ANALYSIS FOR THE ARMY UH-60A IMDS USAGE

by

Ahmet Murat DERE

December 2006

Thesis Advisor:	Samuel E. Buttrey
Second Reader:	Lyn R. Whitaker

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE		Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY	2. REPORT DATE December 2006	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE Flight Regime Recognition Analysis for the Army UH-60A IMDS Usage	5. FUNDING NUMBERS
6. AUTHOR(S) DERE, Ahmet Murat	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) Goodrich Corporation Fuel & Utility Systems, Vergennes, VT 05491	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited	12b. DISTRIBUTION CODE

13. ABSTRACT (maximum 200 words)

Usage Monitoring requires accurate regime recognition. For each regime, there is a usage assigned for each component. For example, the damage accumulated at a component is higher if the aircraft is undergoing a high G maneuver than in level flight.

The objective of this research is to establish regime recognition models using classification algorithms. The data used in the analysis are the parametric data collected by the onboard system and the actual data, consisting of the correct regime collected from the flight cards.

This study uses Rpart (with a tree output) and C5.0 (with a ruleset output) to establish two different models. Before model fitting, the data was divided into smaller datasets that represent regime families by subsetting using important flight parameters. Nonnormal tolerance intervals are constructed on the uninteresting values; then these values in the interval are set to zero to be muted (e.g. excluded). These processes help reduce the effect of noise on classification.

The final models had correct classification rates over 95%. The number of bad misclassifications were minimized (e.g. the number of bad misclassifications of a level flight regime as a hover regime was minimized), but the models were not as powerful in classifying the low-speed regimes as in classifying high-speed regimes.

14. SUBJECT TERMS Flight Regime Recognition, Classification Algorithms,C5.0 ,Rpart, Flight Regime Families	15. NUMBER OF PAGES 127
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

FLIGHT REGIME RECOGNITION ANALYSIS FOR THE ARMY UH-60A IMDS USAGE

Ahmet Murat DERE
First Lieutenant, Turkish Army
B.S., Turkish Military Academy, 1999

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
December 2006

Author: Ahmet Murat DERE

Approved by: Samuel E. Buttrey
Thesis Advisor

Lyn R. Whitaker
Second Reader

James N. Eagle
Chairman, Department of Operations Research

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Usage Monitoring requires accurate regime recognition. For each regime, there is a usage assigned for each component. For example, the damage accumulated at a component is higher if the aircraft is undergoing a high G maneuver than in level flight.

The objective of this research is to establish regime recognition models using classification algorithms. The data used in the analysis are the parametric data collected by the onboard system and the actual data, consisting of the correct regime collected from the flight cards.

This study uses Rpart (with a tree output) and C5.0 (with a ruleset output) to establish two different models. Before model fitting, the data was divided into smaller datasets that represent regime families by subsetting using important flight parameters. Nonnormal tolerance intervals are constructed on the uninteresting values; then these values in the interval are set to zero to be muted (e.g. excluded). These processes help reduce the effect of noise on classification.

The final models had correct classification rates over 95%. The number of bad misclassifications were minimized (e.g. the number of bad misclassification of a level flight regime as a hover regime was minimized), but, the models were not as powerful in classifying the low-speed regimes as in classifying high-speed regimes.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

EXECUTIVE SUMMARY

The Health and Usage Management System (or usage monitoring) determines the actual usage of a component on aircraft. This allows the actual usage from a flight instead of the more conservative worst-case usage to be assigned to that component. By measuring the actual usage on the aircraft, the usage times of components can be extended to their true lifetimes. Usage Monitoring requires an accurate representation of regime (i.e. Regime Recognition). For each regime, there is a usage (a “damage factor”) assigned for each component that has usage For example, the damage accumulated by a component might be higher if the aircraft is undergoing a high-G maneuver than it is during a straight and level flight. These damage factors are assigned by the Original Equipment Manufacturer based on measured stresses in the aircraft when undergoing a given maneuver. Inaccurate regime recognition may lead to a false impression of usage of some aircraft components. This may result in higher cost in maintenance and, more seriously than that, threats to flight safety.

This research establishes two regime recognition models using classification algorithms. The C5.0 classification algorithm was used to produce rulesets and the Rpart package was used to produce tree-based outputs.

The data used in the analysis are the parametric data collected by the onboard system and the actual data, consisting of the correct regime collected from the flight cards. The data for this research was provided by the Goodrich Corporation Fuel & Utility Systems. The data was collected from an experimental flight of an UH-60A “Bearcat5.” The data used in the analysis consists of three parametric data files collected by the onboard system and two flight cards that have the true value of regime and information about the time during which each regime was flown.

Before the model fitting, the data was divided into smaller datasets using important flight parameters. After this preliminary division, nonnormal tolerance

intervals are constructed on the other parameter values to capture the useless information in the data that does not explain anything about regimes. Those nonnormal intervals are constructed in two steps: assuming normality and then revising intervals by visual inspection to compensate for skewness in the data. Values in the intervals are set to zero to be muted (e.g. excluded from modeling process.) Some parameter values are also rounded and transformed without losing useful information. After this data editing process, classification models are fitted to each dataset. Since the variability of the parameters values is low in smaller data subsets, these sub-models are more powerful than any single classification model built on the full data.

The final models had correct classification rates over 95%. The number of bad misclassifications were minimized (e.g. the number of bad misclassification of level flight regimes as hover regimes was minimized), but the models were not as powerful in classifying the low-speed regimes as in classifying high-speed regimes.

I. INTRODUCTION

A. BACKGROUND

In early 1990s, the U.S. military started an aircraft health and usage monitoring integration program aimed at demonstrating and validating emerging technologies. This program is called the Health and Usage Management System (HUMS.) The U.S. military did not have state-of-the-art diagnostic capability installed on rotary-wing aircraft at that time. Based upon the mission need, such a system was expected to enhance operational safety and significantly reduce life cycle cost through its ability to predict impending failure of both structural and dynamic drive system components. This consequently would direct on-condition maintenance actions and/or alert the pilot to conditions affecting flight safety. (Goodrich Corporation, 2001)

The Naval Air Warfare Center Aircraft Division was the pioneer in evaluating diagnostic technologies. The SH-60 was selected as the test vehicle because it offered the best availability of test assets and the highest potential for support because of the large number of aircraft among the Navy, Army and Coast Guard. The program, designated Helicopter Integrated Diagnostic System (HIDS) uses state-of-the-art data acquisition, raw data storage, and algorithmic analysis provided under contract by Goodrich to evaluate the propulsion and power drive system. Cockpit instruments and control positions are recorded during the entire flight for usage monitoring and flight analysis. Since the introduction of structural usage monitoring capability to the HIDS program in 1995, it has led to other joint Goodrich/US Military programs. This capability is expected to provide a significant reduction in maintenance cost while maintaining the current level of safety (Goodrich Corporation, 2001.)

1. System Description

The integrated mechanical diagnostic System (IMDS) includes all of the necessary hardware and software for acquiring data in flight to provide on-aircraft warnings and maintenance advisories. The system also includes a separate

ground station that performs post-flight analysis, data processing, maintenance diagnostics, reporting, and data archiving. The ground station hardware and software are designed to be operable in the current U.S. Navy/USCG/U.S. Marine Corps maintenance environment and provide maintenance data output products that can be readily integrated with the Navy's maintenance concept and daily operation.

A regime is a category of operation that an aircraft can be in at a given time. A regime can also be known as a flight condition or maneuver. The usage monitoring subsystem determines the percentage of flight time the helicopter has spent in each flight regime as well as the regime sequence (i.e., flight profile). The regime data is then used to calculate the rate at which various structural components are being used up, and when they need to be removed from service so as to maintain the required reliability (Goodrich Corporation, 2002a.)

a. On-Board System

The airborne portion of the system includes interfaces to sensors, signal conditioning and data acquisition capability for all sensors, and the algorithms required to complete all of the in-flight functions and the data transfer the ground station.

b. Central Ground-Based System

The parameter data is downloaded to the Central Ground-Based Station (CGBS) after each flight. Regimes are then recognized using the downloaded data, and a usage spectrum is generated based upon the regime sequence. The major functionality of the ground-based station is to communicate usage reports to the server. One of these reports is the “structural life limited parts usage report.” This usage report reports the usage accumulated during a single flight (Goodrich Corporation, 2001.) Figure 1 shows photographs of the MPU (Main Processing Unit) and CDU (Central Processing Unit), and Figure 2 shows a screen shot of a flight summary in the ground station.

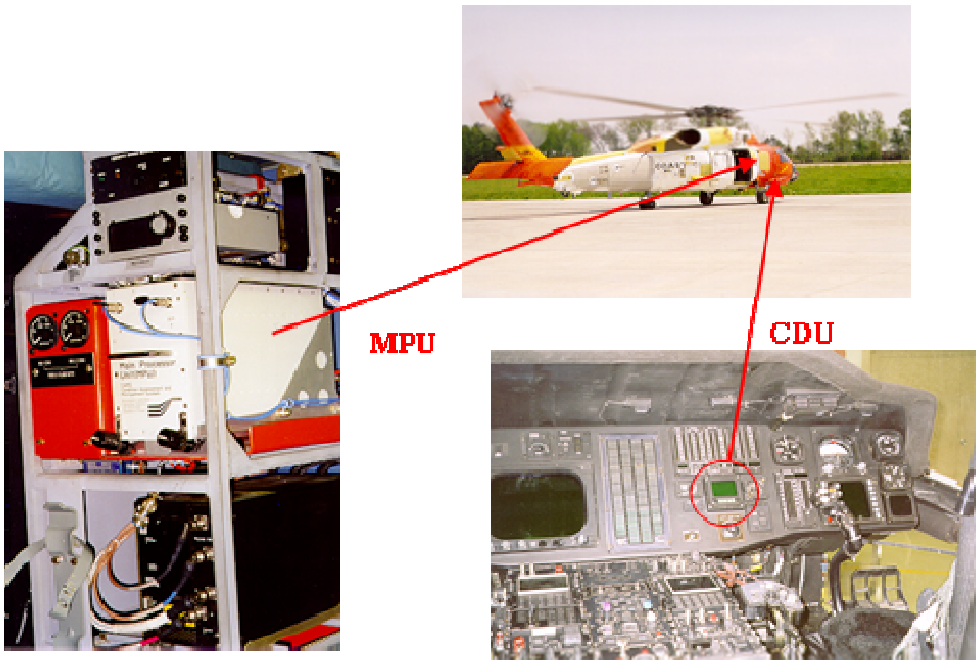


Figure 1. HUMS On-Board System Installed on a Helicopter (From Goodrich, 2001)

Figure 2. An Example of the Ground Station Screen of a Flight Summary and Flight Spectrum Report for a Single Flight (From Goodrich, 2001)

B. OBJECTIVE

Usage Monitoring requires accurate representation of regime (i.e. Regime Recognition). For each regime, there is a usage assigned for each component. For example, the damage accumulated at a component is higher if the aircraft is undergoing a high-G maneuver than it is during a straight and level flight. These damage factors are assigned by the Original Equipment Manufacturer based on measured stresses in the aircraft when undergoing a given maneuver (Bechhoefer, n.d.)

The objective of this thesis research is to establish a regime recognition model using classification algorithms. The data used in the analysis is the parametric data collected by the onboard system and the actual data from flight cards which has the exact information on the flight. The data was provided by the Goodrich Corporation Fuel & Utility Systems. The data was collected from an experimental flight of an UH-60A "Bearcat5." The data consists of three parametric data files collected by the onboard system and two flight cards that have the true value of regime and information about the time during which each regime was flown. The model based on this data should minimize the number of bad classifications (e.g. no classification of a level-flight regime as a hover regime)

C. SCOPE

The Health and Usage Management System determines the actual usage of a component on the aircraft. This allows the actual usage from a flight instead of the more conservative worst-case usage to be assigned to that component. By measuring the actual usage on the aircraft, the life of components can be extended to their true lifetime (Bechhoefer, n.d.) This is directly related to the accurate representation of regime recognition. Inaccurate regime recognition may lead to a false impression of usage of some aircraft components. This may result in higher cost in maintenance and, more seriously than that, threats to flight safety.

D. ORGANIZATION OF THESIS

This thesis is comprised of five chapters. Chapter II focuses on the previous studies on this research topic. Chapter III gives short descriptions of the parameters of the data and the process of preparing the data for model fitting. Chapter IV gives an overview of the possible models and algorithms and also explains how and why the best model is chosen. Chapter V summarizes the result of the models and presents recommendations for future studies.

THIS PAGE INTENTIONALLY LEFT BLANK

II. PREVIOUS STUDIES

A. PREVIOUS STUDIES ON THE SAME DATASET BY THE GOODRICH CORPORATION

1. The Maximum Likelihood Estimator Methodology

This previous work was by Eric Bechhoefer, Goodrich Corporation Fuel & Utility Systems, using a maximum likelihood estimator (MLE) methodology. MLEs assume that input parameters are noisy, and weight the validity of a parameter by its system variance. The output of the algorithm is the regime which is most likely, as a function of the a priori parameter variance. That is, the technique measures the difference between the observed parameters and those from a notional set of regimes. The regime which is closest, statistically, to the measured parameters, is most likely. In fact, the MLE is a multi-dimensional hypothesis test, in which the parameters are used to test the hypothesis that the current set of parameters is a member of a given regime (Bechhoefer, n.d.)

2. The Logical Tests

Before the MLE Methodology, the Goodrich Corporation was using a logical test-based regime recognition model. The specific parameter cases are tested and if the test result is true for a regime, that regime was considered true. This method would have been a good methodology, if the measured parameters were free of noise. However, many of the parameters used for regime recognition models are noisy. This may lead to a large number of misclassifications. This thesis research establishes a regime recognition model based on classification algorithms and presents different ways to reduce the effect of noise in the data by subsetting, transforming and filtering.

3. The Approach in the Current Research

The approach of this thesis research is to use classification algorithms to establish a classification model with rulesets or classification trees that ensures a minimal number of bad classifications. The first step is to fit different models to the training set using different classification algorithms. The algorithm that gives the best results is chosen for further modeling. This further modeling seeks

different ways to improve the classification rate and also focuses on fixing modeling problems. To achieve this goal, before fitting the best model, the dataset was partitioned based on values of three parameters. Partitioning was performed first using weight on wheels, and then calibrated airspeed and then control reversals. There was an additional splitting for only C5.0 models using the take-off / landing parameter. This process yields smaller datasets that represent families of regimes. For example, the dataset where weight on wheels is “0” and calibrated airspeed is less than a selected threshold value will be called the “in the air and slow” family. This family is comprised of the observations when the aircraft is in the air and at low speed. A sub-model was fit to the dataset of each family. Due to the low variability of the parameter values in these small subsets (families), the sub-models are more powerful than any single classification model fit to the whole dataset. The sub-models are established using parameters which are considered to have important information about that regime family. One reason for the small number of bad misclassifications is that important parameters are used for the preliminary partitioning process. Besides this preliminary classification process, a filtering methodology (i.e. muting the uninteresting values of some parameters) is applied to prevent potential problems caused by noise in the parameter values. This process consists of building non-normal tolerance intervals on some selected parameters and rounding the parameter values without losing interesting information. The non-normal intervals were built starting with the normality assumption and then revised by visual inspection. Data in these revised intervals was set to zero to mute those values so that the algorithm never thinks that they are interesting enough to split on. Only extreme values that carry important information about the regime are of interest. This process also prevents splits on small and uninteresting values of the input parameters.

B. OTHER RELATED STUDIES

1. Regime recognition for MH-47E Structural Usage Monitoring

This logical test-based study was done by Boeing Defense & Space Group, Helicopters Division (1997.)

They obtain high quality flight data measurements through the use of data editing and data filtering techniques, to define the maneuver state of the aircraft in terms of a comprehensive set of fundamental maneuvers, and to determine the MH-47E basic fatigue profile flight regime which best describes the maneuver state of the aircraft (Teal et al., 1997.)

Data conditioning, filtering, and failure management techniques were primarily used. Wind direction and magnitude estimation and inertial/air data blending were applied to obtain high-fidelity airspeed estimation at low speeds. Maneuver identification algorithms and criteria were presented and validated using flight test data. As a result, the methodology they used for mapping the aircraft maneuver state into the MH47E Basic Fatigue Profile flight regimes ensured a conservative, yet realistic, assessment of critical component life expenditure.

THIS PAGE INTENTIONALLY LEFT BLANK

III. DATA

A. DATA USED IN THE ANALYSIS

1. The Definitions of the Parameters from the Parametric Data Files

This portion of the data used in the regime recognition analysis is from various aircraft state parameters collected by the on-board system for usage monitoring. The parameter definitions are taken from (Goodrich Corporation, 2002b), (UH-60A Operator's Manual, 1996) and (Wikipedia, 2006). Some of the plots illustrate unexpected behaviors in the data; others are just for visualization of the parameters. For a better visualization of the role of the parameters in different flight regimes, only one plot is shown for each parameter. Each plot uses the parameter values from a regime in which that parameter is important in determining that regime.

a. Airspeed.Vh.Fraction

This continuous parameter is the ratio of the actual speed to the speed achieved in level flight with maximum continuous power. In Figure 3, the x-axis shows the time frame in seconds during which a level-flight regime is flown and the y-axis shows the corresponding values of this parameter. For this level-flight regime, *Airspeed.Vh.Fraction* is a very important parameter and its values should be observed in the planned interval of 0.3 and 0.4. The planned values are in fact observed in that interval of time for this level-flight regime.

b. Altitude.Rate

This continuous parameter is the vertical velocity of the aircraft in feet per minute. In Figure 4, the x-axis shows the time frame in seconds during which a right climbing turn regime is flown and the y-axis shows the corresponding values of the parameter *Altitude.Rate*. For this flight regime, *Altitude.Rate* is a very important parameter and it should achieve positive and gradually increasing values since the aircraft is gaining altitude. The expected values are in fact observed in that interval of time for this climbing regime.

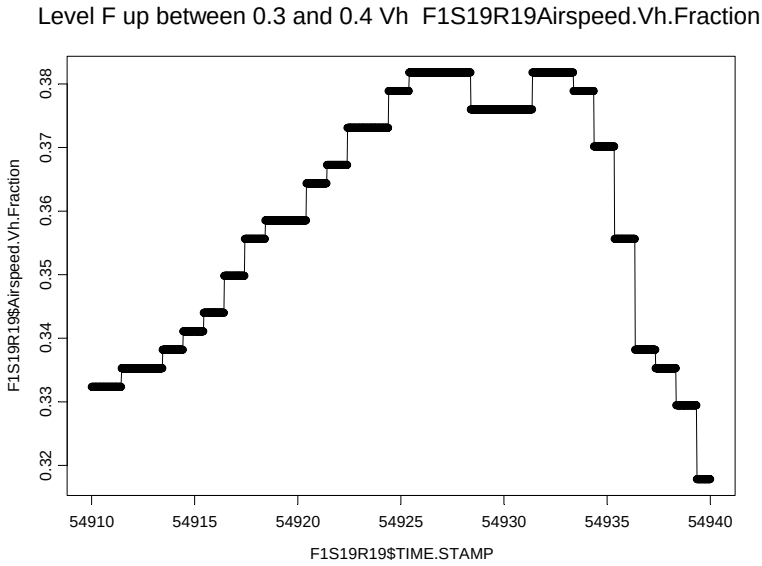


Figure 3. Airspeed.Vh.Fraction in a Level-flight

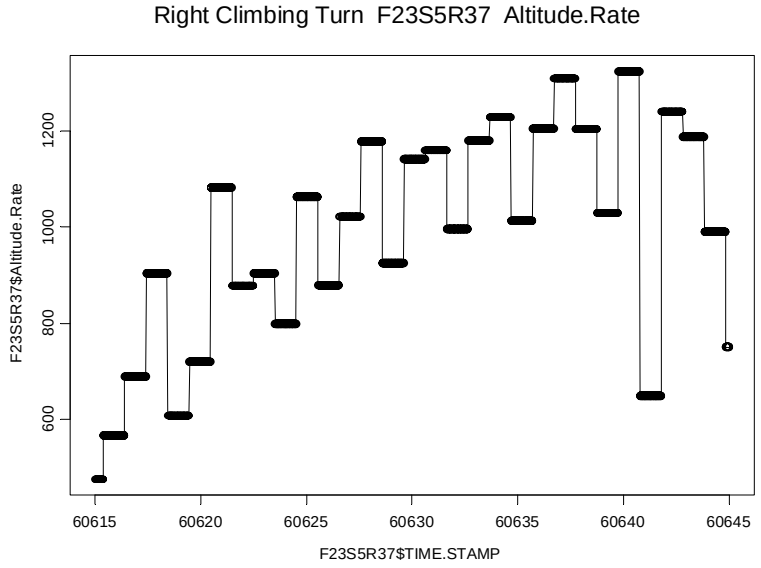


Figure 4. Altitude.Rate in Right Climbing Turn Regime.

c. Angle.of.Bank

This continuous parameter is the angle (in degrees) between the aircraft's normal axis (longitudinal axis) and the vertical plane. *Angle.of.Bank* is the absolute value of the parameter *Roll.Attitude*. In the left plot in Figure 5, the x-axis shows the time frame in seconds during which a right turn regime is flown and the y-axis shows the corresponding values of the parameter *Angle.of.Bank*. In the right plot in Figure 5, the x-axis shows the time frame in seconds during which a level flight regime is flown and the y-axis shows the corresponding values of the parameter *Angle.of.Bank*. The flat line is the expected angle of bank from the flight card; the other line is the observed angle of bank from the parametric data files. The plot on the right in Figure 5 shows that in response to helicopter movement, there will be a small angle of bank observed, even if the expected angle of bank is zero.

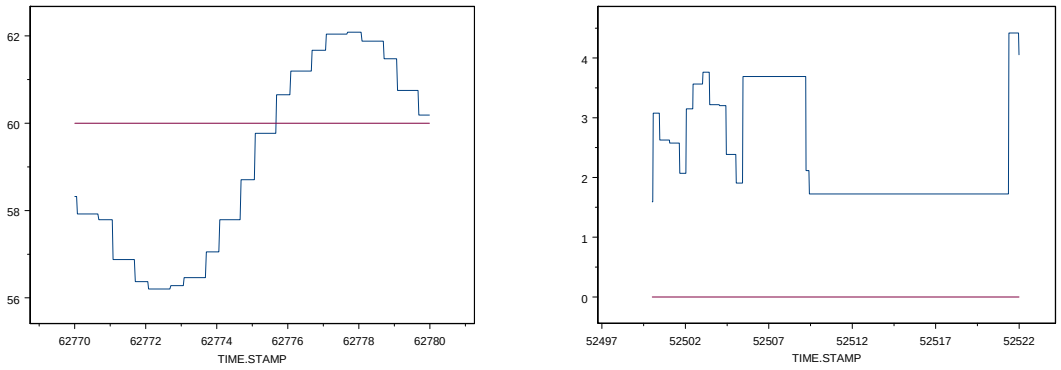


Figure 5. The Angle.of.Bank in Turn Right with 60° Max AOB.\and in Level-flight with 0° AOB

d. KCAS

This continuous parameter illustrates indicated airspeed, corrected for instrument error and position error (see also parameter *KIAS*). In the following plot, the x-axis shows the time frame in seconds during which a level-flight regime is flown and the y-axis shows the corresponding values of the parameter

KCAS. For this level-flight regime the parameters values should be observed in the approximate interval of 70 to 90. The planned values are in fact observed in that interval of time for this level-flight regime.

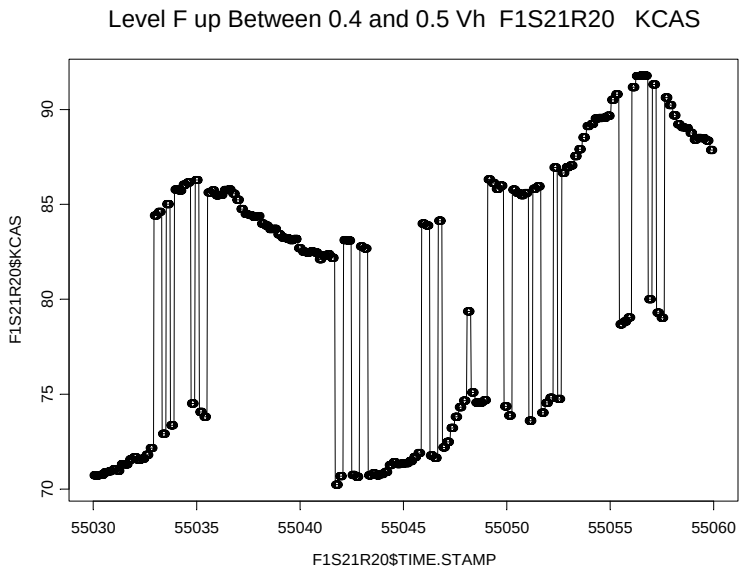


Figure 6. KCAS in Level Flight up between 0.4 and 0.5 Vh Regime.

e. CONTROL.REVERSAL.ID

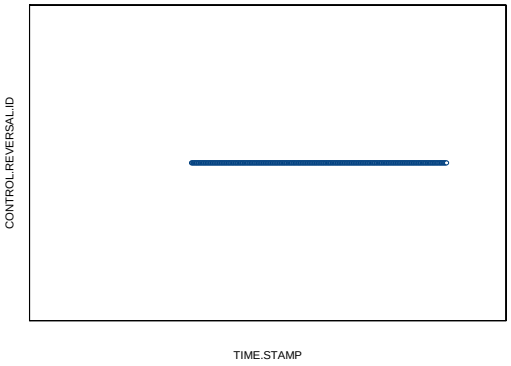
This categorical parameter is a control input made by the pilot to maintain the current position of the aircraft, normally during a wind gust. It can also be a pilot control input made for evasive maneuvers. The reversal consists of control input in one direction and a reversal to return the control to its starting position. This will be in the form of a number in the set of {0, 1, 2, 4, 8}, as defined in the chart below. This will last as long as the control reversal is present.

Number	Longitudinal Cyclic Reversal Present	Lateral Cyclic Reversal Present	Pedal Reversal Present	Collective Reversal Present
0	No	No	No	No
1	No	No	No	Yes
2	No	No	Yes	No
4	No	Yes	No	No
8	Yes	No	No	No

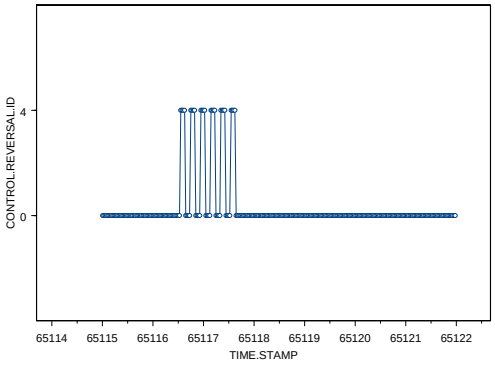
Table 1. Control Reversal IDs (Goodrich, 2002a)

Some regimes are directly related to this parameter. The parameter should be observed with its expected ID level. The charts below show the regimes, expected IDs and observed IDs. The parameter is plotted as if it were continuous.

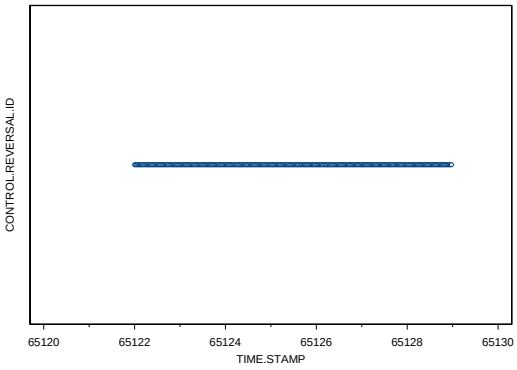
REG15 Rudder Reversal in Hover expected ID=2



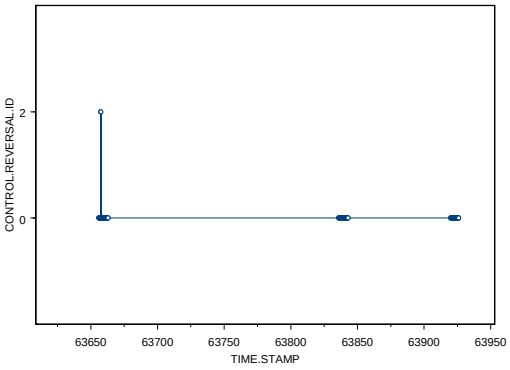
REG16 Longitudinal Reversal in Hover expected ID=8



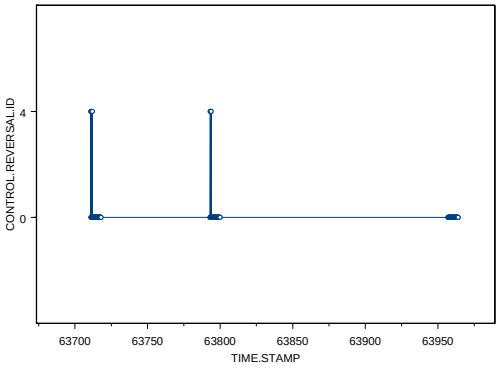
REG17 Lateral Reversal in Hover expected ID=4



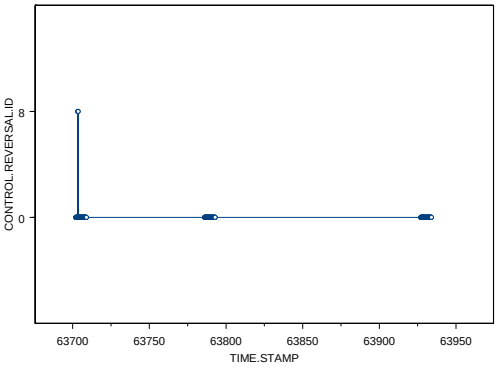
REG26 Rudder Reversal in Level Flight to 1.0 Vh expected ID=2



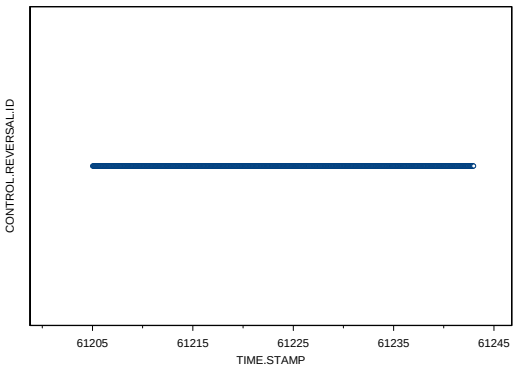
REG27 Lateral Reversal in Level Flight to 1.0 Vh expected ID=4



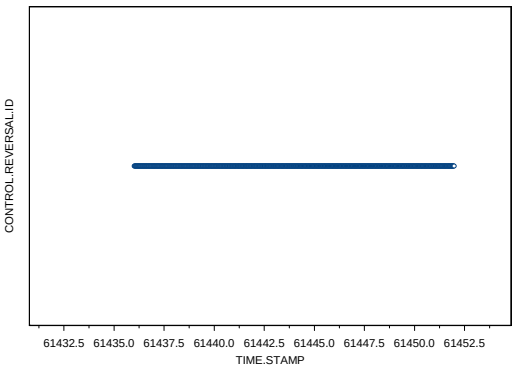
REG28 Longitudinal Reversal in Level Flight to 1.0 Vh expected ID=8



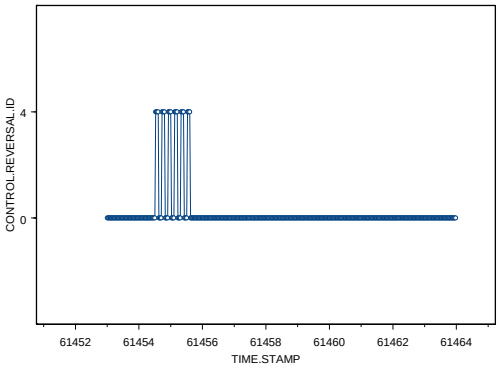
REG43 Rudder Reversal in Autorotation expected ID=2



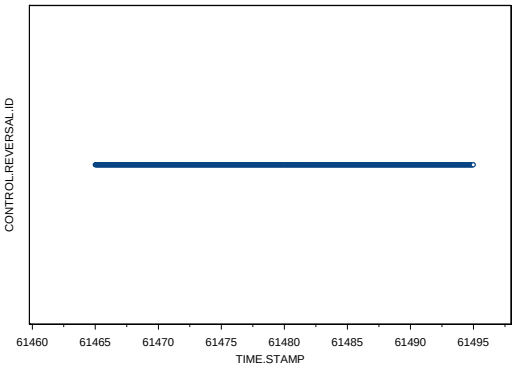
REG44 Longitudinal Reversal in Autorotation expected ID=8



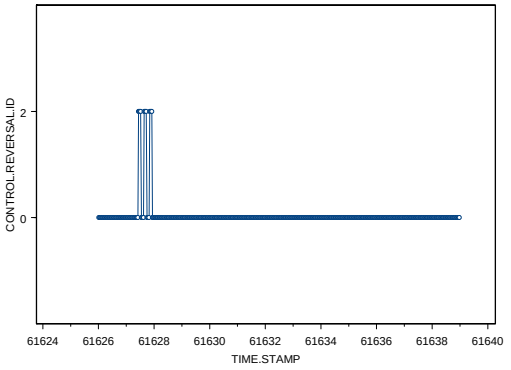
REG45 Lateral Reversal in Autorotation expected ID=4



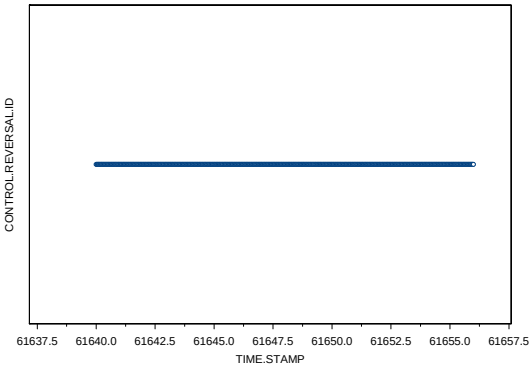
REG46 Collective Reversal in Autorotation expected ID=1



REG48 Rudder Reversal in Partial Power Descent expected ID=2



REG49 Longitudinal Reversal in Partial Power Descent expected ID=8



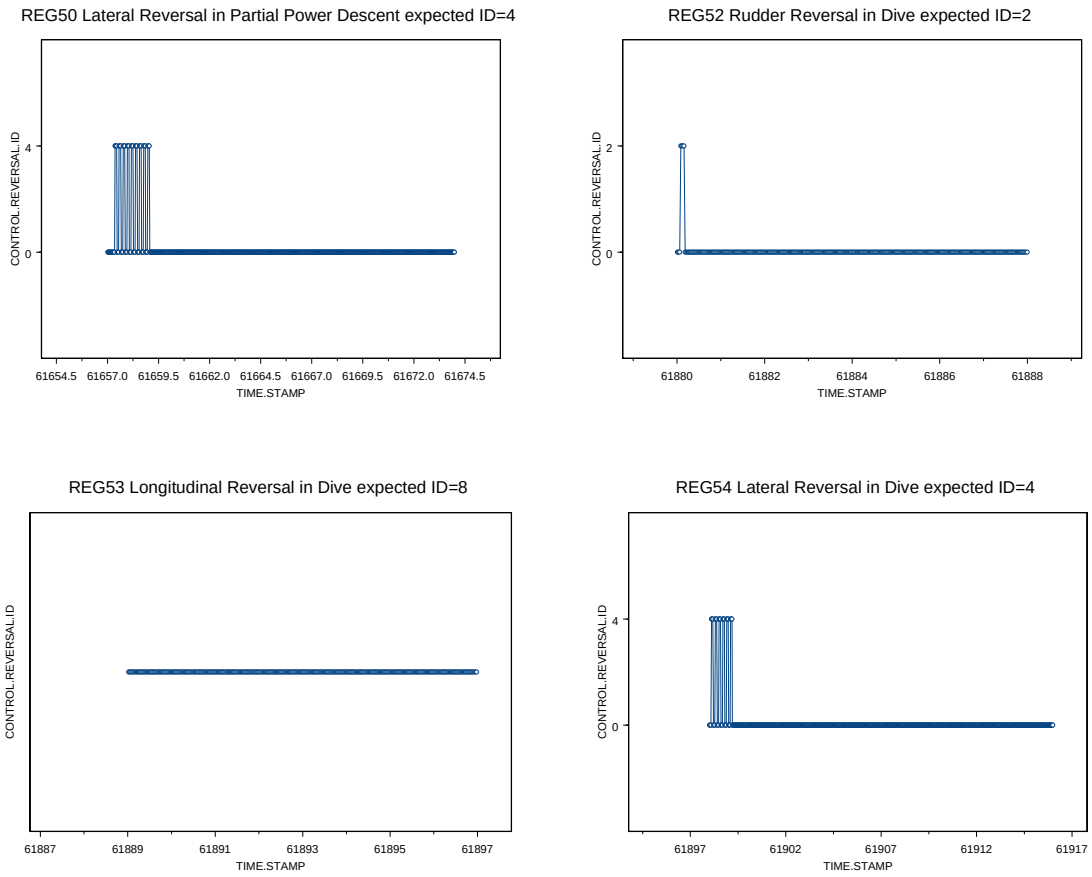


Figure 7. Control Reversal IDs.

The expected IDs are observed in eight cases; in one case, the wrong level of ID is observed and in seven cases no IDs are observed. Since the reversals should require a maximum of two seconds, the on-board system might not be capturing the command inputs by the pilot. Since the preliminary classification process, which subsets data into smaller regime families, uses this parameter, those unobserved ID levels cause misclassification by directing observations into undesired regime families.

f. Landing Flag

This categorical parameter represents aircraft landing, i.e., if the wheels are in contact with the ground after not being in contact with the ground is 1; otherwise it is 0. This is a very important parameter. If it is 1, it should be possible to assume that the regime is a landing one, without ever inspecting the

other parameters. *Landing.Flag* gets the value of 1 when *Weight.On.Wheels* transitions from 0 to 1, and *Weight.On.Wheels* remains 1 for one second. *Landing.Flag* is set back to 0 by the system after five seconds of being 1.

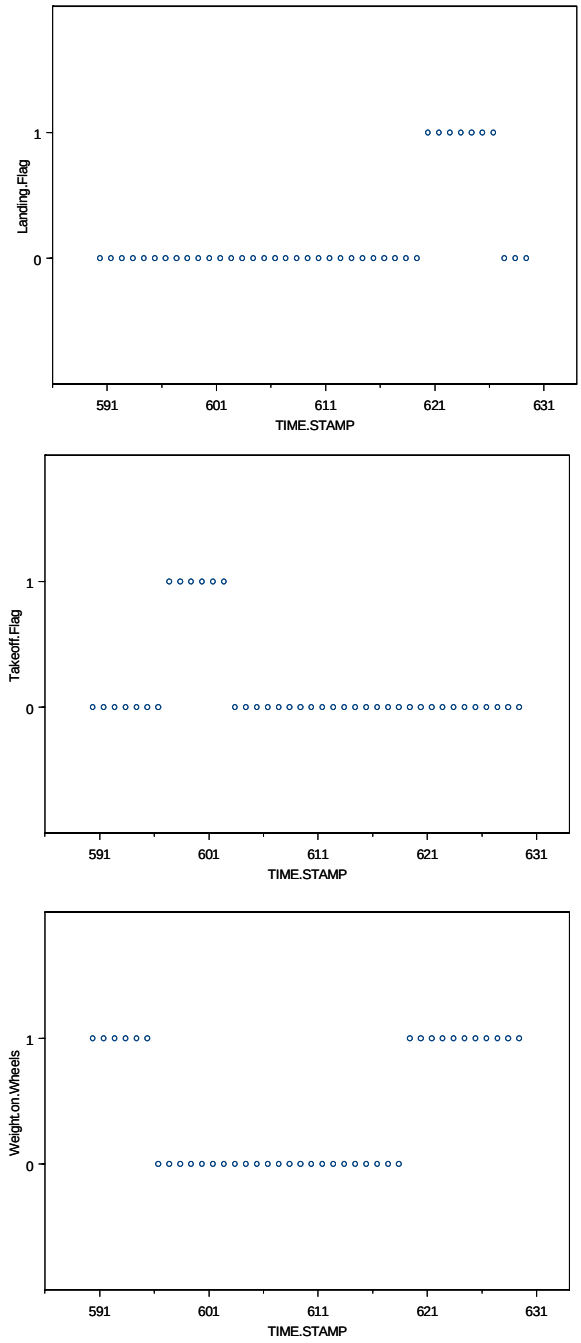


Figure 8. Weight.On.Wheels, Takeoff.Flag and Landing Flag

In the plot above, the transition of *Weight.On.Wheels* from 0 to 1 sets *Landing.Flag* to 1 and after approximately five seconds, the system sets it back to 0.

g. Takeoff. Flag

This categorical parameter represents aircraft take-off, i.e., if the wheels are not in contact with the ground it is 1; otherwise it is 0. This is a very important parameter. If it is 1, it should be possible to assume that the regime is in takeoff without ever inspecting the other parameters. In the following plot, the x-axis shows the time frame in seconds that a take-off regime is flown and the y-axis shows the corresponding values of the parameter *Takeoff.Flag*.

Takeoff.Flag gets the value of 1 when *Weight.On.Wheels* transitions from 1 to 0, and *Weight.On.Wheels* remains 0 for one second. *Takeoff.Flag* is set back to 0 by the system, after five seconds of being 1. In the plot below, the transition of *Weight.On.Wheels* from 1 to 0 sets *Takeoff.Flag* to 1 and after approximately five seconds, the system sets it back to 0.

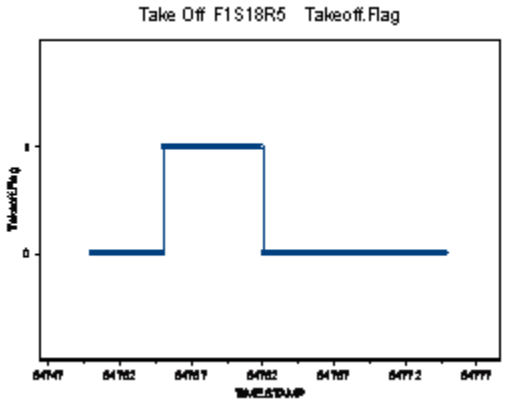


Figure 9. Takeoff.Flag in a Take-off Regime.

Since a period where the parameter is “1” means a take-off regime, the rest of the times where *Takeoff.Flag* is “0” (in that particular picture) may not be predicted as being in a take-off regime. Those observations will become misclassifications since the corresponding responses for these observations (in that time period) are all take-off regimes.

g. Weight.on.Wheels

This categorical parameter is 1 if the aircraft is on ground; if the aircraft is in flight then its value is “0”. This parameter adds delay and keeps regime recognition in synchronization. By the system, the landing and takeoff parameters are delayed for approximately two seconds after this parameter, in order to prevent falsely detected landings and takeoffs.

h. Lateral.Accel

This continuous parameter is the lateral acceleration of the aircraft in G's. Acceleration in the port direction is positive. In the following plot, the x-axis shows the time frame in seconds during which a left turn regime is flown and the y-axis shows the corresponding values of the parameter *Lateral.Accel*. In a left turn regime, the parameter values should achieve positive values since the left is the port side. The expected behavior is observed.

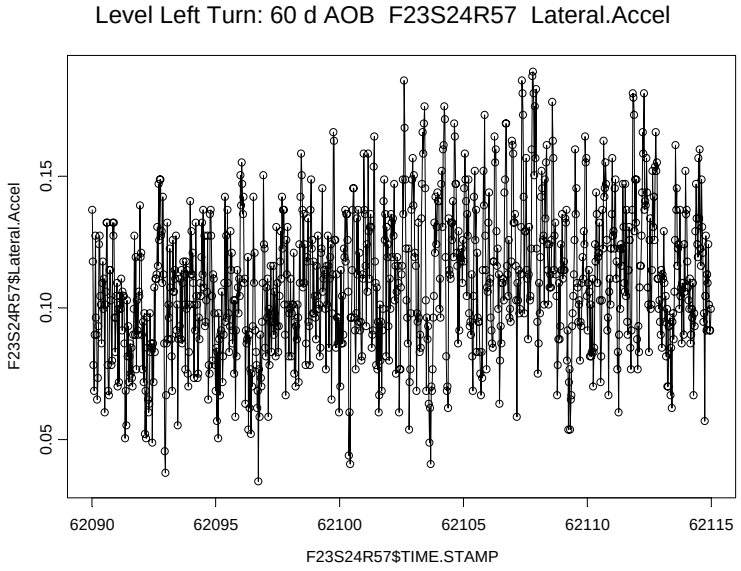


Figure 10. Lateral.Accel in a 60-degree Left Turn Regime.

i. Nr

This continuous parameter is the main rotor RPM or rotor rate of rotation in percent. In the following plot, the x-axis shows the time frame in seconds during which a take-off regime is flown and the y-axis shows the corresponding values of the parameter *Nr*. The increasing part of the curve represents the actual time when the aircraft becomes airborne.

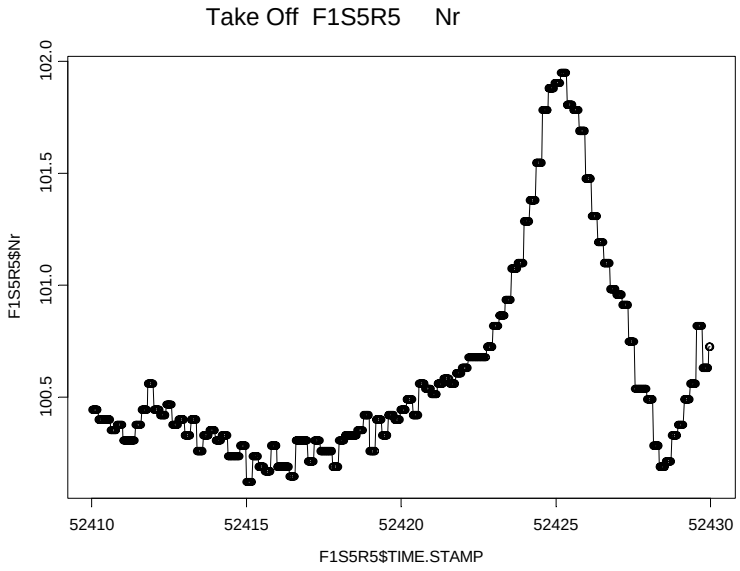


Figure 11. Nr in a Take-off Regime.

j. Pitch.Attitude

This continuous parameter is pitch angle in degrees. If the aircraft's nose is up, the parameter value is positive. In an unloaded helicopter, the nose tends to move up, which causes this parameter to be positive in level and slow flights. In Figure 12, the x-axis shows the time frame in seconds during which a rearward flight regime is flown and the y-axis shows the corresponding values of the parameter *Pitch.Attitude*. For this flight regime, this parameter is a very important. The parameters values should be positive values to achieve a nose up flight pattern.

k. Radar.Altitude

This continuous parameter is the instantaneous indication of actual terrain clearance height in feet. In Figure 13, the x-axis shows the time frame in seconds during which a hover regime is flown and the y-axis shows the corresponding values of the parameter *Radar.Altitude*. In an in-ground-effect hover regime, this parameter values should achieve less than 80 feet, and this can be readily observed in the plot.

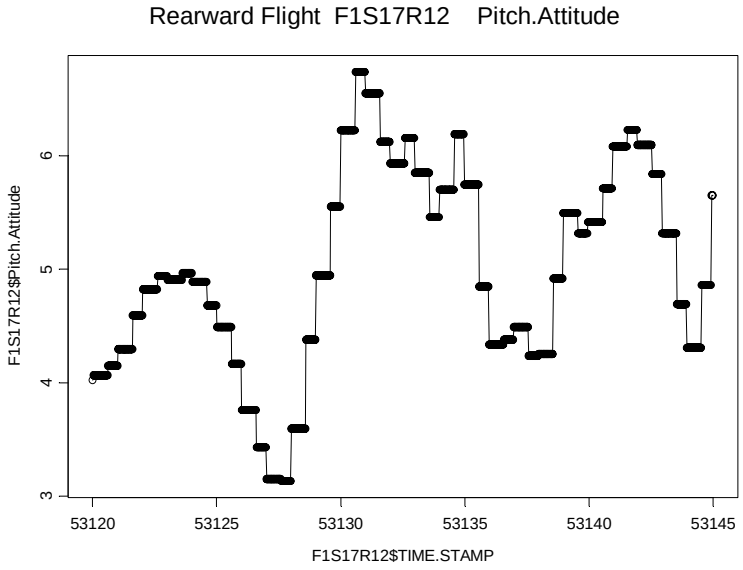


Figure 12. Pitch.Attitude in a Rearward Flight Regime.

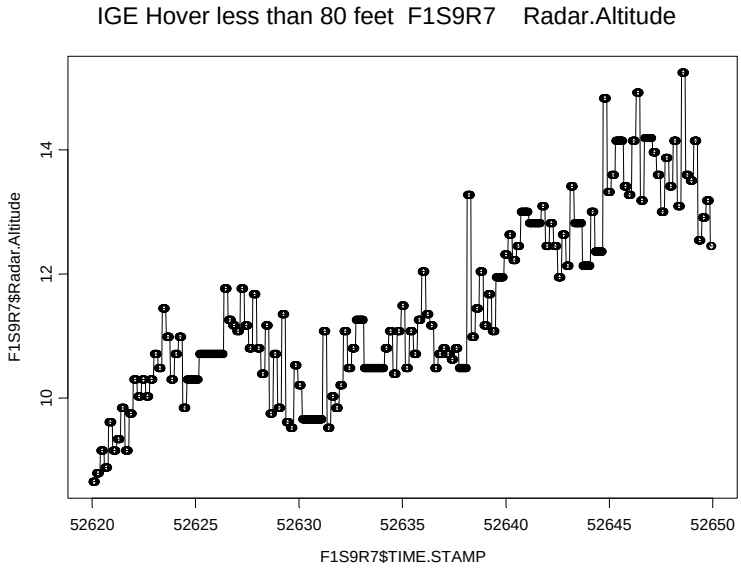


Figure 13. Radar.Altitude in an In-Ground-Effect-Hover Regime.

l. Roll. Attitude

This continuous parameter is the roll angle in degrees. If the aircraft has a right bank, the parameter values are positive. In the following plot, the x-axis shows the time frame in seconds during which a descending left turn regime is flown and the y-axis shows the corresponding values of the parameter *Roll. Attitude*. This descending regime also consists of a banked turn with a 60° angle of bank. Since this 60°-bank is a left bank, it should achieve negative values. This pattern can be readily observed in Figure 14.

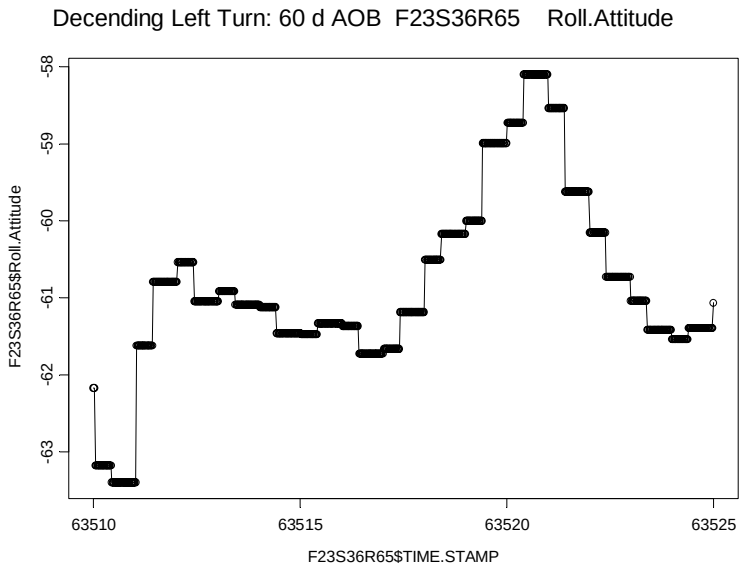


Figure 14. Roll.Attitude a Descending Left Turn 60° Max AOB Regime.

m. TGT.1 and TGT.2

These continuous parameters are the turbine gas temperature in engine 1 and engine 2 respectively. The units are in °C. In the left plot below, the x-axis shows the time frame in seconds during which a hover regime is flown and the y-axis shows the corresponding values of the parameter TGT.1 and in the right one, the x-axis shows *TGT.1* and the y-axis shows *TGT.2*. The approximate linear relationship between these two parameters can be readily observed. Therefore, *TGT.1* and *TGT.2* were combined into a parameter called *TGT* by averaging *TGT.1* and *TGT.2.*

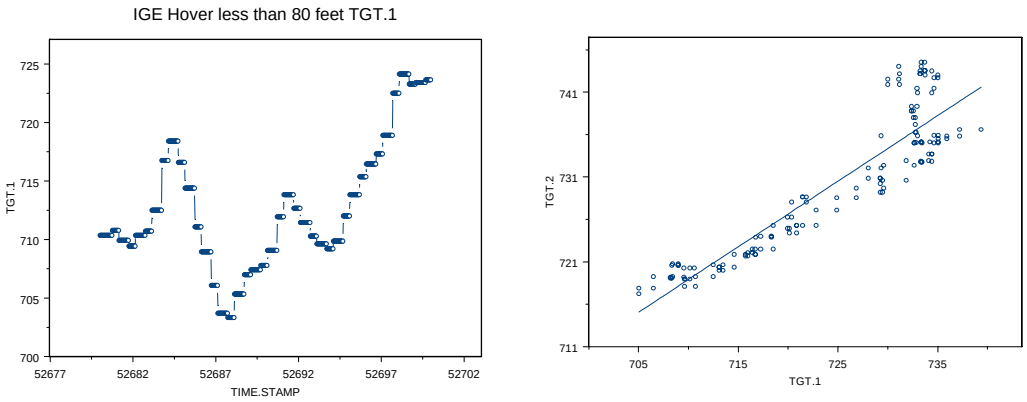


Figure 15. TGT.1 values in a Hover Regime (the plot on the right shows the relationship between TGT.1 and TGT.2.)

n. Torque.1 and Torque.2

These continuous parameters are torque acquired from engine 1 and engine 2 respectively. In the right plot below, the x-axis shows the time frame in seconds during which a take-off regime is flown and the y-axis shows the corresponding values of the parameter *Torque.1*.In the left plot below shows *Torque.1* vs *Torque.2*. The approximate linear relationship between these two parameters can be readily observed. Therefore, *Torque.1* and *Torque.2* will be combined into a parameter called *Torque* by averaging *Torque.1* and *Torque.2.*

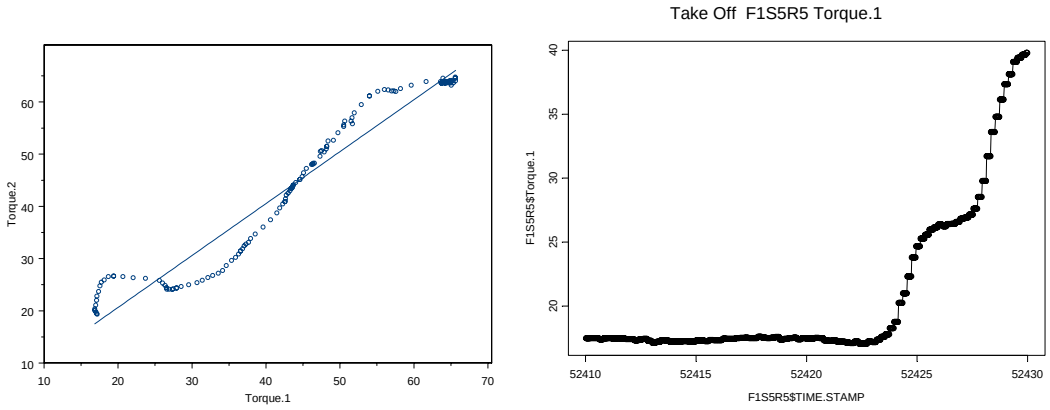


Figure 16. Torque.1 vs. Torque.2 and Torque.1 in a Take-off Regime.

o. Vertical.Accel

This continuous parameter is the vertical acceleration of the aircraft in G's. If the aircraft's acceleration is in the up direction, the parameter is positive. In the following plot, the x-axis shows the time frame in seconds during which a level right turn regime is flown and the y-axis shows the corresponding values of the parameter *Vertical.Accel*.

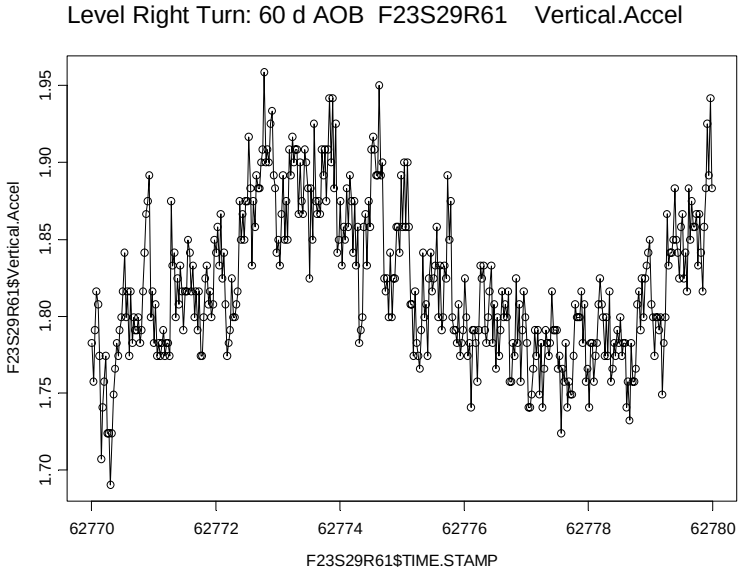


Figure 17. Vertical.Accel in a Level Right Turn with a 60-degree-angle of bank.

In the plot above, the values of *Vertical.Accel* are in the interval of 1.7 to 1.95 (and close to 2). Since the flight regime is a level flight, it is expected to observe 1. The main reason for this undesired behavior is that the level turn includes a 60-degree-angle of bank. This negatively affects stability in a level flight. While executing this maneuver, a collective input by the pilot is required to maintain the stability of the aircraft which causes the aircraft to gain some altitude in that a period of time. This results in a G bigger than expected.

p. Yawrate

This continuous parameter is the change of yaw in degrees per second (°/s). If it is positive, it means increasing; negative indicates decreasing. In the following plots, the x-axes show the time frame in seconds during which

hover turns are flown and the y-axes show the corresponding values of the parameter *Yawrate*.

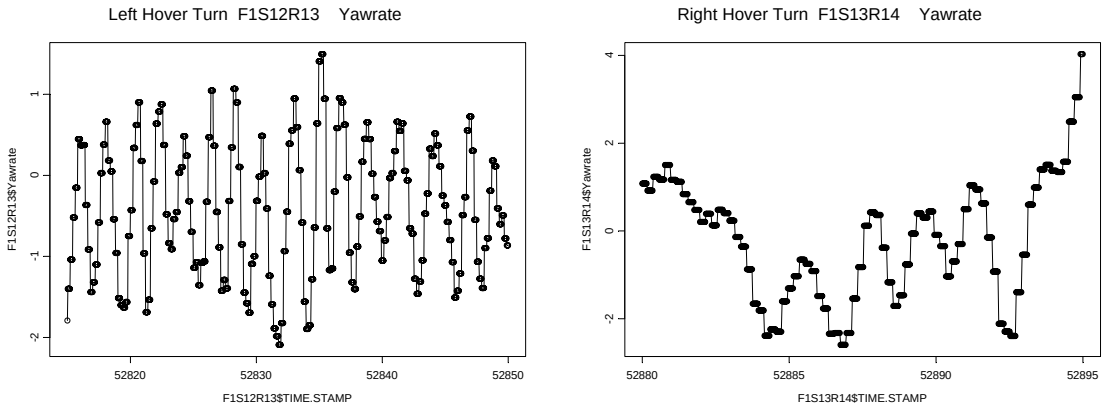


Figure 18. Yawrate in Hover Turns.

The plot above left shows the parameter values for a left hover turn regime. If the aircraft is making a left turn in hover, the yaw angle should achieve negative values. Therefore, it is expected that most of the time the values of the parameter *Yawrate* would be negative, but they are not. The plot on the right shows a right hover turn regime. If the aircraft is making a right turn in hover, the yaw angle should achieve positive values. Therefore, it is expected that most of the time the values of the parameter *Yawrate* would be positive, but they are not. These behaviors may cause misclassification.

2. The Definitions of the Parameters from the Actual Flight Cards

This portion of the data used in the regime recognition analysis comes from actual flight cards. The actual flight cards have flight information on the actual (or expected) levels of some parameters. The aircraft Bearcat 5 was told to execute aircraft maneuvers which were planned in the flight cards. These cards also have the value of the regime that would be realized if the flight were carried out as planned. The times on the flight card were used to map and create subsets from the parametric data files for each regime. Consequently, all of the available parameters from the flight cards were added to the regime data sets.

The parameter definitions are from (Goodrich Corporation, 2002b), (UH-60A Operator's Manual, 1996) and (Wikipedia, 2006.)

Table 2 shows a sample portion from a flight card. For example, between the given time interval from 16:39:37 to 16:40:07, the aircraft was in regime 36. The speed read from the display was 80 knots. The pressure altitude was 3000 feet. .The planned angle of bank was 15 degress. The rate of climb was 1000 ft/min and there was no control reversal planned for this time interval.

hour	minute	sec	KIAS	Palt	AOB	RC	CR	Regime
16	39	37	80	3000	15	1000		36
16	40	7	80	3000				
16	41	25	90	3000	0	-500		40
16	41	55	90	3000				

Table 2. A Sample Portion of a Flight Card (From Goodrich Documents)

a. KIAS

This continuous parameter is the speed read directly from the airspeed indicator on an aircraft, driven by the pitot-static system. KIAS is directly related to knots calibrated airspeed (*KCAS*), but includes instrument errors and position error (See *KCAS.*)

b. Palt

This continuous parameter indicates the pressure altitude measured above sea level on a standard atmospheric day.

c. RC

This continuous parameter indicates the expected rate of climb, which is the speed at which an aircraft increases its altitude expressed in feet per minute (ft/min).This parameter is directly related to the altitude rate.

d. AOB

This continuous parameter is the expected angle of bank (See *Angle.of.Bank.*)

e. CR

This categorical parameter is the expected control reversal ID (See *Control.Reversal.ID.*)

f. Regime

This categorical parameter is the number of the regime that the aircraft is in. A regime is a category of operation that an aircraft can be in at a given time. An aircraft can only perform in a single regime at a time. A regime can also be known as a flight condition (See Table 3.)

Regime	Regime Name	Regime	Regime Name
2	Power On Aircraft, Rotors Turning, Taxi or Stationary	36	Left Climbing Turn
3	left Taxi Turn	37	Right Climbing Turn
4	Right Taxi Turn	40	Autorotation
5	Take Off	41	Autorotation with Left Sideslip
7	IGE Hover less than 80 feet	42	Autorotation with Right Sideslip
8	0GE Hover greater than 80 feet	43	Rudder Reversal in Autorotation
9	Fwd Flight to 0.3 Vh	44	Longitudinal Reversal in Autorotation
10	Right Sideward Flight	45	Lateral Reversal in Autorotation
11	Left Sideward Flight	46	Collective Reversal in Autorotation
12	Rearward Flight	48	Rudder Reversal in Partial Power Descent
13	Left Hover Turn	49	Longitudinal Reversal in Partial Power Descent
14	Right Hover Turn	50	Lateral Reversal in Partial Power Descent
15	Rudder Reversal in Hover	51	Dive
16	Longitudinal Reversal in Hover	52	Rubber Reversal in Dive
17	Lateral Reversal in Hover	53	Longitudinal Reversal in Dive
19	Level Flight up between 0.3 and 0.4 Vh	54	Lateral Reversal in Dive
20	Level Flight up Between 0.4 and 0.5 Vh	55	Level Left Turn: 30 d AOB
21	Level Flight up Between 0.5 and 0.6 Vh	56	Level Left Turn: 45 d AOB
22	Level Flight up Between 0.6 and 0.7 Vh	57	Level Left Turn: 60 d AOB
23	Level Flight up Between 0.7 and 0.8 Vh	59	Level Right Turn: 30 d AOB
24	Level Flight up Between 0.8 and 0.9 Vh	60	Level Right Turn: 45 d AOB
25	Level Flight up Between 0.9 and 1.0 Vh	61	Level Right Turn: 60 d AOB
26	Rudder Reversal in Level Flight to 1.0 Vh	63	Decending Left Turn: 30 d AOB
27	Lateral Reversal in Level Flight to 1.0 Vh	64	Decending Left Turn: 45 d AOB
28	Longitudinal Reversal in Level Flight to 1.0 Vh	65	Decending Left Turn: 60 d AOB

Table 3. Regimes (showing only the ones present in the dataset) (From Goodrich, 2002b)

3. The Derived Parameters

In order to lower the dimensionality of the data, some parameters were combined into one parameter when appropriate.

a. TGT

(See *TGT.1* and *TGT.2*)

b. Torque

(See *Torque.1* and *Torque.2*)

B. DATA EDITING PROCESS FOR MODEL FITTING

Initial fitting of classification models using the C5.0 algorithm lead to models which split on noisy variables and which were difficult to interpret. To avoid this, several of the parameters were muted using the data editing process in this section.

1. Building Tolerance Intervals Assuming Normality

This section addresses an issue that is very important for the regime recognition model. This issue is data editing, or the removal of uninteresting values of some parameters by muting or giving them default values, so that the model would never think that those values are interesting enough to build structures on (i.e. split on them in a classification model.) Different regimes can be observed when some parameters happen to have bigger or more extreme values than usual. If the usual values are muted by giving them a value of 0, the extreme values, which help to define regimes, will be more obvious and therefore they will be more likely to be captured by the model. This may help avoid having unnecessary splitting on noisy values when the classification models are applied. It may also contribute to the interpretability of the models.

Not all parameters are important to all regimes. Yaw rate is a very descriptive parameter for a hover left turn regime, whereas pitch attitude is not a very important one for that specific regime. If unusual values were observed for the pitch attitude in a left hover regime, then the classification might no longer be left hover turn. For example, if the pitch attitude happens to be positive and larger

than a threshold, then this regime would be classified as a rearward flight regime. If an interval of unimportant (usual) values of the unimportant parameters is transformed into uninteresting values such as “0,” then the descriptive (important) parameters will drive the classification model and result in high correct classification rates. In doing so, the interpretability of the model also increases. A rough idea that some parameters have usual values which do not explain anything about the regime was extracted from Goodrich Documents. This idea was then converted into the importance matrices given in this section

The data is divided into three parts: on the ground, in the air and fast, and in the air and slow. This process is critical because of the different stability conditions for different aircraft regimes. If the aircraft is on the ground, it is hard to expect big changes in the roll attitude and pitch attitude; these values will be in a narrow interval. If the aircraft is flying, its stability will be less, which will make those acceptable intervals wider. Even in the fast and slow speed states, these parameters will behave differently from one another. More speed may mean more deviation in parameter values.

For data editing, only altitude rate, pitch attitude, yaw rate, roll attitude and angle of bank were selected. These parameters are very important in most of the regimes and they explain a great deal about the basics of a flight regime.

The purpose is to find an interval for the selected parameters where they are not explaining anything about the regime and to set values in those intervals to “0.” Table 4 illustrates how to decide where those parameters are not important.

	If the parameter is important for regime then the value is 1.	Altitude.Rate	Pitch.Attitude	Yawrate	Roll.Attitude	Angle.of.Bank
2	Power On Aircraft, Rotors Turning, Taxi or Stationary	0	0	0	0	0
3	Left Taxi Turn	0	0	**1**	0	0
4	Right Taxi Turn	0	0	**1**	0	0
5	Take-Off	0	0	0	0	0

Table 4. The Importance Matrix for “On The Ground” Regimes

Parameters which have “0” in the matrix are taken from the corresponding regime data files and merged into a vector. This vector consists of the values of those parameter values which have no role in explaining anything about any regime.

	If the parameter is important for regime then the value is 1.	Altitude.Rate	Pitch.Attitude	Yawrate	Roll.Attitude	Angle.of.Bank
7	IGE Hover less than 80 feet	0	0	0	0	0
8	0GE Hover greater than 80 feet	0	0	0	0	0
9	Fwd Flight to 0.3 Vh	0	0	0	0	0
10	Right Sideward Flight	0	0	0	**1**	0
11	Left Sideward Flight	0	0	0	**1**	0
12	Rearward Flight	0	**1**	0	0	0
13	Left Hover Turn	0	0	**1**	0	0
14	Right Hover Turn	0	0	**1**	0	0
15	Rudder Reversal in Hover	0	0	0	0	0
16	Longitudinal Reversal in Hover	0	0	0	0	0
17	Lateral Reversal in Hover	0	0	0	0	0

Table 5. The Importance matrix for “In The Air and Slow (hover)” regimes

To form an interval of uninteresting values, a first step might be to assume Normality for the parameters so that the usual tolerance intervals can be constructed (Devore, 2004.) Since some parameters will not be normally distributed, the interval will be checked on the scatter plot of the parameter. Skewness and nonnormality will affect the interval. The symmetric tolerance interval is calculated under the normality assumption and then will be revised by this visual inspection. Checking the scatter plots to revise the intervals compensates for not incorporating the skewness into the interval calculations.

The tolerance interval should be as narrow as possible to capture as much uninterestingness as possible without touching the interesting part. For this purpose, only about 68% of the values will be captured by the tolerance interval. This percentage is close to +/-1 standard error of the data.

	If the parameter is important for regime then the value is 1.	Altitude.Rate	Pitch.Attitude	Yawrate	Roll.Attitude	Angle.of.Bank
19	Level Flight up between 0.3 and 0.4 Vh	0	0	0	0	0
20	Level Flight up Between 0.4 and 0.5 Vh	0	0	0	0	0
21	Level Flight up Between 0.5 and 0.6 Vh	0	0	0	0	0
22	Level Flight up Between 0.6 and 0.7 Vh	0	0	0	0	0
23	Level Flight up Between 0.7 and 0.8 Vh	0	0	0	0	0
24	Level Flight up Between 0.8 and 0.9 Vh	0	0	0	0	0
25	Level Flight up Between 0.9 and 1.0 Vh	0	0	0	0	0
26	Rudder Reversal in Level Flight to 1.0 Vh	0	0	0	0	0
27	Lateral Reversal in Level Flight to 1.0 Vh	0	0	0	0	0
28	Longitudinal Reversal in Level Flight to 1.0 Vh	0	0	0	0	0
36	Left Climbing Turn	**1**	0	0	**1**	0
37	Right Climbing Turn	**1**	0	0	**1**	0
40	Autorotation	**1**	**1**	0	0	0
41	Autorotation with Left Sideslip	**1**	**1**	0	0	0
42	Autorotation with Right Sideslip	**1**	**1**	0	0	0
43	Rudder Reversal in Autorotation	**1**	**1**	0	0	0
44	Longitudinal Reversal in Autorotation	**1**	**1**	0	0	0
45	Lateral Reversal in Autorotation	**1**	**1**	0	0	0
46	Collective Reversal in Autorotation	**1**	**1**	0	0	0
48	Rudder Reversal in Partial Power Descent	**1**	**1**	0	0	0
49	Longitudinal Reversal in Partial Power Descent	**1**	**1**	0	0	0
50	Lateral Reversal in Partial Power Descent	**1**	**1**	0	0	0
51	Dive	**1**	**1**	0	0	0
52	Rudder Reversal in Dive	**1**	**1**	0	0	0
53	Longitudinal Reversal in Dive	**1**	**1**	0	0	0
54	Lateral Reversal in Dive	**1**	**1**	0	0	0
55	Level Left Turn: 30 d AOB	0	0	0	**1**	**1**
56	Level Left Turn: 45 d AOB	0	0	0	**1**	**1**
57	Level Left Turn: 60 d AOB	0	0	0	**1**	**1**
59	Level Right Turn: 30 d AOB	0	0	0	**1**	**1**
60	Level Right Turn: 45 d AOB	0	0	0	**1**	**1**
61	Level Right Turn: 60 d AOB	0	0	0	**1**	**1**
63	Descending Left Turn: 30 d AOB	**1**	0	0	**1**	**1**
64	Descending Left Turn: 45 d AOB	**1**	0	0	**1**	**1**
65	Descending Left Turn: 60 d AOB	**1**	0	0	**1**	**1**

Table 6. The Importance Matrix For “In The Air and Fast Regimes”

TOLERANCE INTERVALS FOR IN THE AIR AND FAST					
	AltRate	AOB	Pitch.Att	Roll.Att	YawRate
Lower	-617.1	-5.1	-4.1	-2.7	-6.5
Upper	477.5	17.3	4.2	2.7	3.4

TOLERANCE INTERVALS FOR IN THE AIR AND SLOW					
	AltRate	AOB	Pitch.Att	Roll.Att	YawRate
Lower	-314.4	1.1	1.8	-3.9	-2.8
Upper	320.9	3.6	5.4	-1.4	1.8

Table 7. The Calculated Tolerance Intervals

If the values are between the intervals given above, they will be set to zero.

2. Revising the Calculated Intervals Due to Nonnormality

In this section, the calculated intervals were compared to the actual scatter plots to revise them due to the parameters' skewed and nonnormal characteristics.

In the scatter plots, the dotted lines show the lower and the upper values for the calculated tolerance intervals. The solid lines show the revised values. The dotted lines are moved up or down to find a region that can separate the values very distinctively. The objective is to find a region for the solid lines where they exclude uninteresting information, while including useful information.

As expected that this process helps the classification (C5.0 and Rpart) algorithms identify useful information. The algorithms produces rule sets that are easy to interpret and use.

The revised interval for *AltRate* is [-500,500] (See Figure 19.) The normality assumption is not bad for this parameter (See Figure 20). The left tail is a little heavier than the right one. This is reasonable since even in level flight the aircraft would lose altitude most of the time rather than gaining altitude.

In Figure 19, the dotted lines are the calculated intervals, and the interval lines are moved to an area where they can achieve a better distinct cut-off value of uninteresting values. The cut-off values form the revised intervals where they are used to exclude uninteresting information.

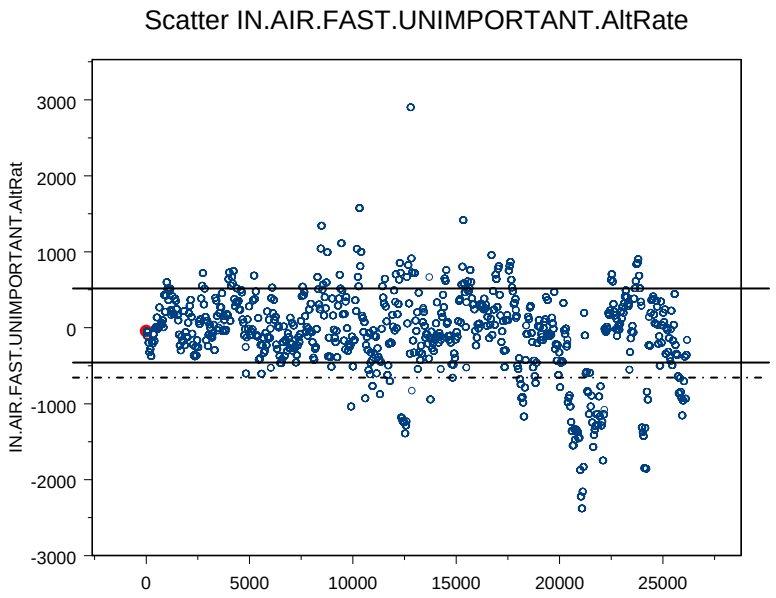


Figure 19. The Revised AltRate interval

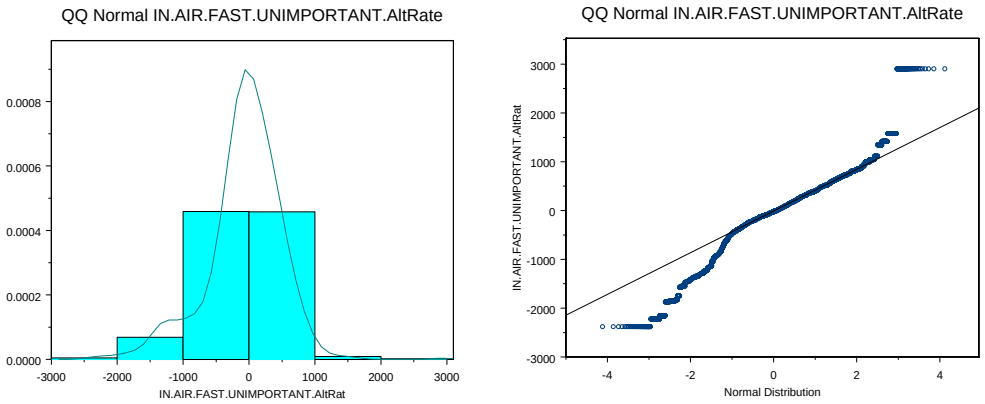


Figure 20. The Histogram and QQ Plot for AltRate

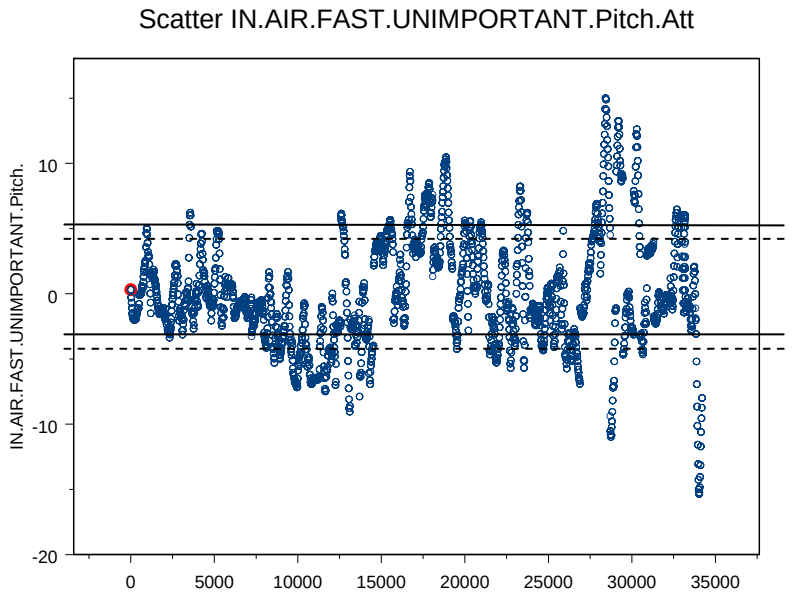


Figure 21. The Revised Pitch.Att Interval

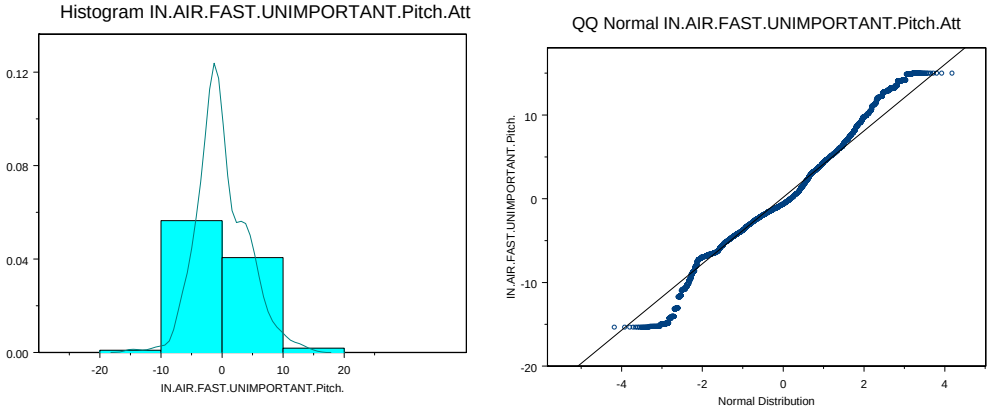


Figure 22. The Histogram and QQ Normal Plot for Pitch.Att

The revised interval for the *Pitch.att* is [-3,5].

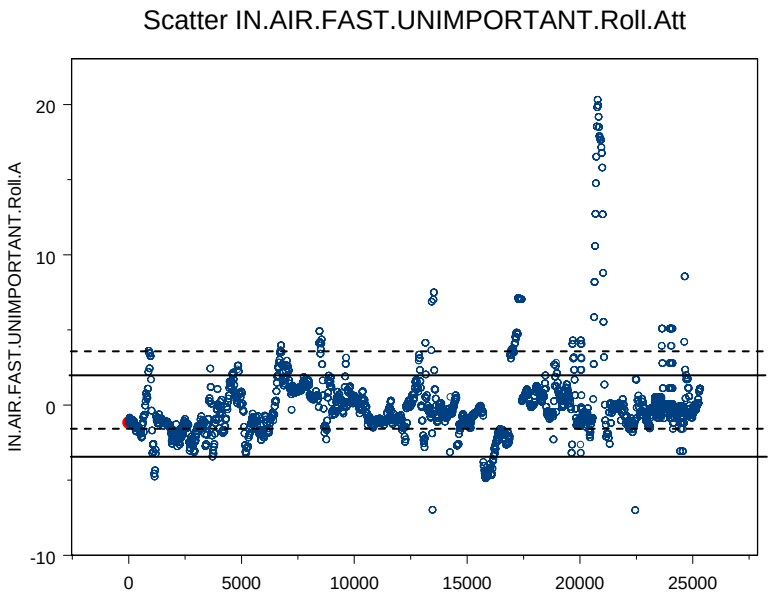


Figure 23. The Revised Roll.Att Interval

The revision for the *Roll.Att* is [-4, 1]. In the plot above the dotted lines are the calculated intervals; the interval lines are moved to a threshold value where they can achieve a better distinct cut-off value.

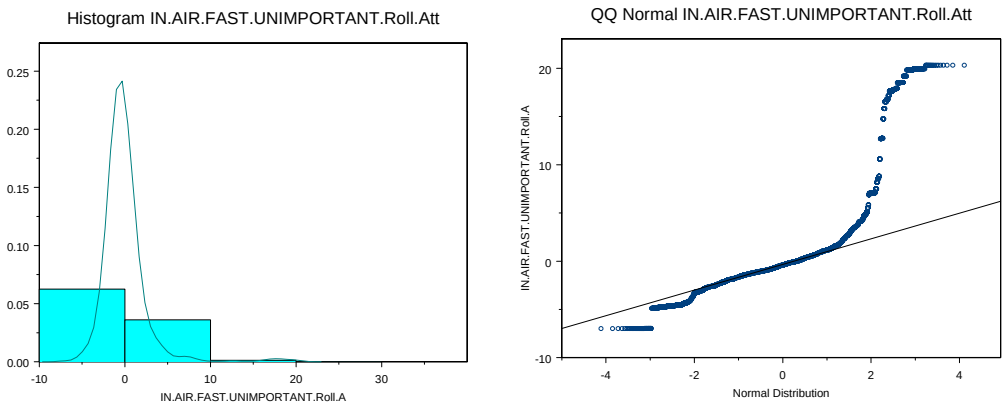


Figure 24. The Histogram and QQ Normal Plot for Roll.Att

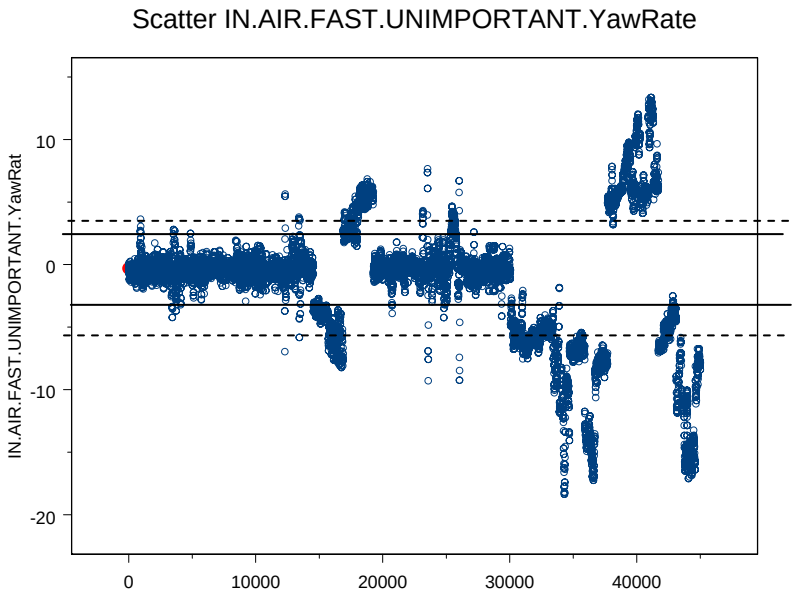


Figure 25. The Revised Yaw Rate Interval

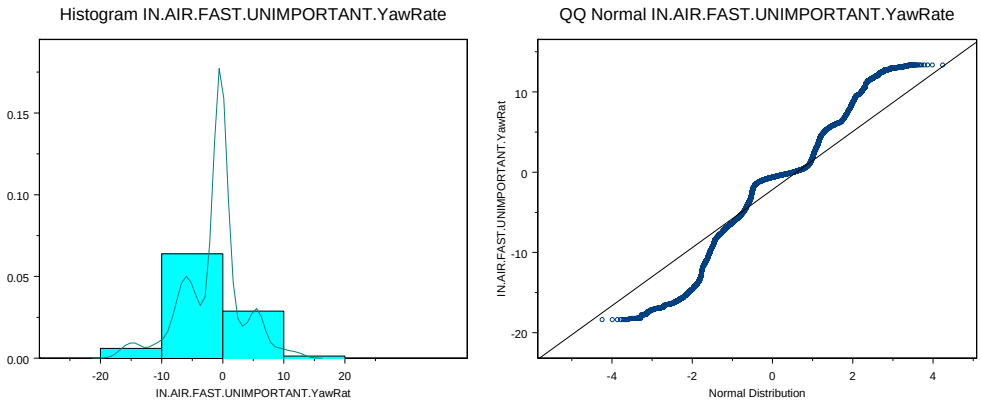


Figure 26. The Histogram and QQ Normal Plot for YawRate

The normality assumption is not very bad for *YawRate*, but the distribution looks multimodal. The revised interval for the *YawRate* is [-3, 2].

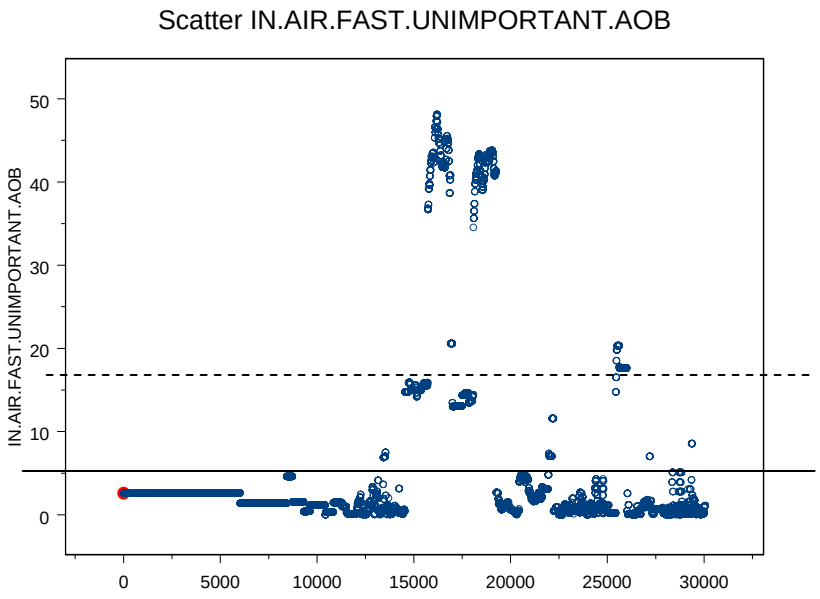


Figure 27. The Revised AOB Interval

Since the assumption of normality does not hold for the roll attitude, it will not hold for the angle of bank. Without checking for normality, the interval for angle of bank will be revised. The revised interval for the AOB is [0, 5].

The same procedure is applied to the in the air and slow data set; the revised and the calculated interval values are close. (See Figures 29-30.)

REVISED INTERVALS FOR IN THE AIR AND FAST					
	AltRate	AOB	Pitch.Att	Roll.Att	YawRate
Lower	-500	0	-3	-4	-3
Upper	500	5	5	1	2

REVISED INTERVALS FOR IN THE AIR AND SLOW					
	AltRate	AOB	Pitch.Att	Roll.Att	YawRate
Lower	-300	0	2	-4	-2
Upper	300	5	5	-2	2

Table 8. The Revised Intervals

3. Rounding the Values of the Selected Parameters

To reduce the variability of the parameter values, some parameters are rounded. When rounding it is very important not to lose useful information. Splits on those parameters are inspected before and after the rounding process to make sure that this process did not mask the information that could define a regime better. This process also helps by giving splitting for continuous parameters which are also rounded. For the on-the-ground data set, parameters values were not rounded, since the rounding process did not make a difference in classification.

Rounding Process	
Parameter	Decimal Place
Airspeed.Vh.Fraction	1
Roll.Attitude	0
Altitude.Rate	0
Pitch.Attitude	0
Yawrate	0
Vertical.Accel	1
Lateral.Accel	1
Radar.Altitude	0
Nr	1

Table 9. The Rounded Parameters and Their Decimal Places

4. Making the Number of Observations Equal for Each Regime in Training &Test Sets

The data that will be used for analysis are from the experimental flight for regime recognition. More data was collected in some regime than in others. For example, the observations for the IGE Hover regime are more numerous than any other regime. The regimes that contain a reversal have a very small number of observations. In actual flight during normal operations, the distribution of the time spent in each of the regimes might differ considerably from the data used here.

Directly partitioning the data into training and test set will cause the distribution of number of observations in different regimes in the data to be carried to the training and test sets. This may force the algorithms to focus on predicting the most frequent regime in the sample. This can be prevented by

making the number of observations equal for all regimes, in both the training and test sets. Besides, different base rates are not of interest for the study; therefore, a uniform distribution of regimes should be used by making the number of observations equal for all possible classes.

The other important issue to consider is that some algorithms may not have the ability to incorporate changes in the distributions of the number of observations in the different regimes. They may use default priors such as “1/#regimes” which will affect the classification. Or they may have alternative ways of accounting for different distributions (e.g. using misclassification costs.)

For these reasons, the numbers of observations for each regime were made equal. The regime data sets were split into subsets from the big flight data using the regime flight times given in the flight card. Since there are multiple flights for some regimes, data sets that belong to the same regime are merged into one data set. Some of the regime data sets are big, but some of them are not. To make the number of observations equal for all data sets, 500 was chosen as the base number of observations. If a data set had more than the base number, only 500 observations were sampled from that data set. Here sampling does not mean randomly choosing some observations without replacement; it means one in every “ #obs in data set /500 ” was taken. This guarantees that any possible pattern in the data set will be carried to the new “smaller” dataset. If a data set has fewer observations than the base number then a sampling with replacement is done until the base number is reached (See Figure 28 and 29).

At this point, all of the regime dataset files had the same number of observations. The next step is to divide each of the regime data sets into training and test sets. Two-thirds of each data set was used for training and the rest as a test set. Selection for test and training sets is done by using an “evenly distributed” approach again, so that all patterns are included in both sets. Whether or not this approach is useful in capturing all possible patterns in the data can be easily checked by inspecting the levels of the *Control.Reversal.ID* parameter in both the test and training sets. Since *Control.Reversal.ID* changes

for a very small amount of time, such as two seconds or so, if the change in the parameter is observed both in the test and training set then the partitioning is successful. This process helps the algorithm to focus on that potential pattern in order to be able to recognize it in the prediction process.

In Figure 28, it is quite evident that the levels of *Control.Reversal.ID* go both to the training and test set. The approximate proportion of the number of observations in different sets can be surmised by looking at the thickness of the plots. In the last step, all of the small data sets are merged into large training and test sets.

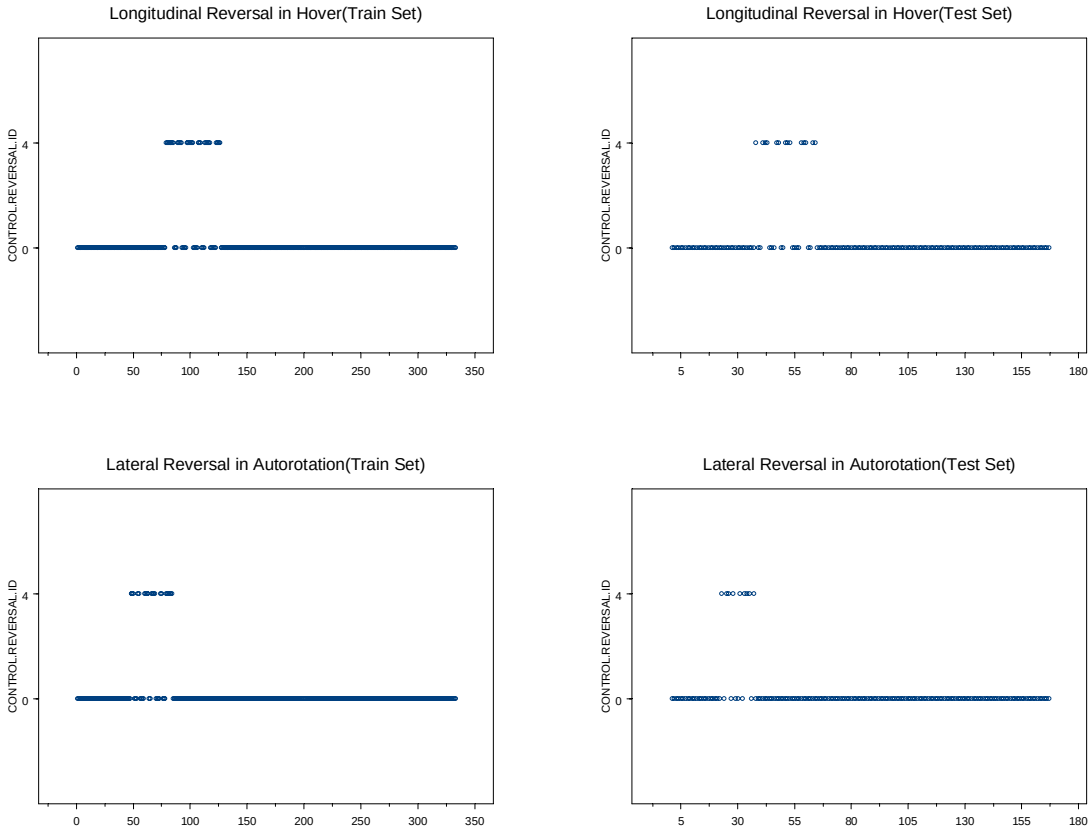


Figure 28. The Presence of *Control.Reversal.IDs* in Both the Training and Test set

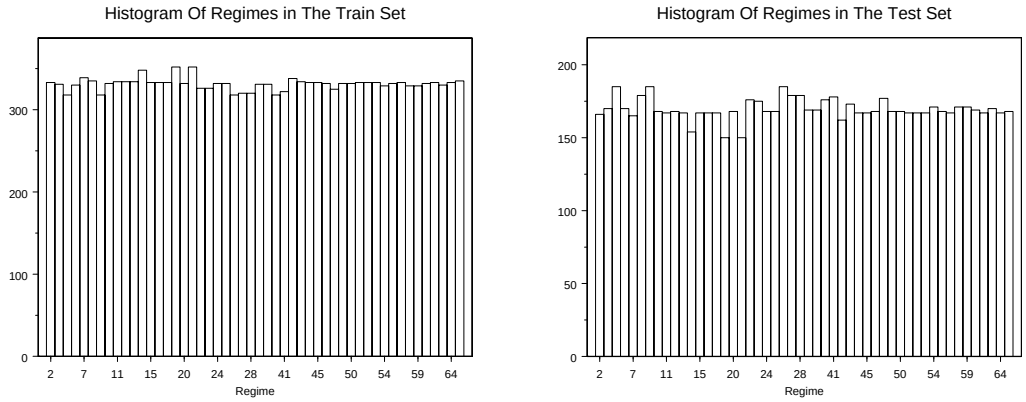


Figure 29. Histograms of the Regimes in the Training/Test Sets

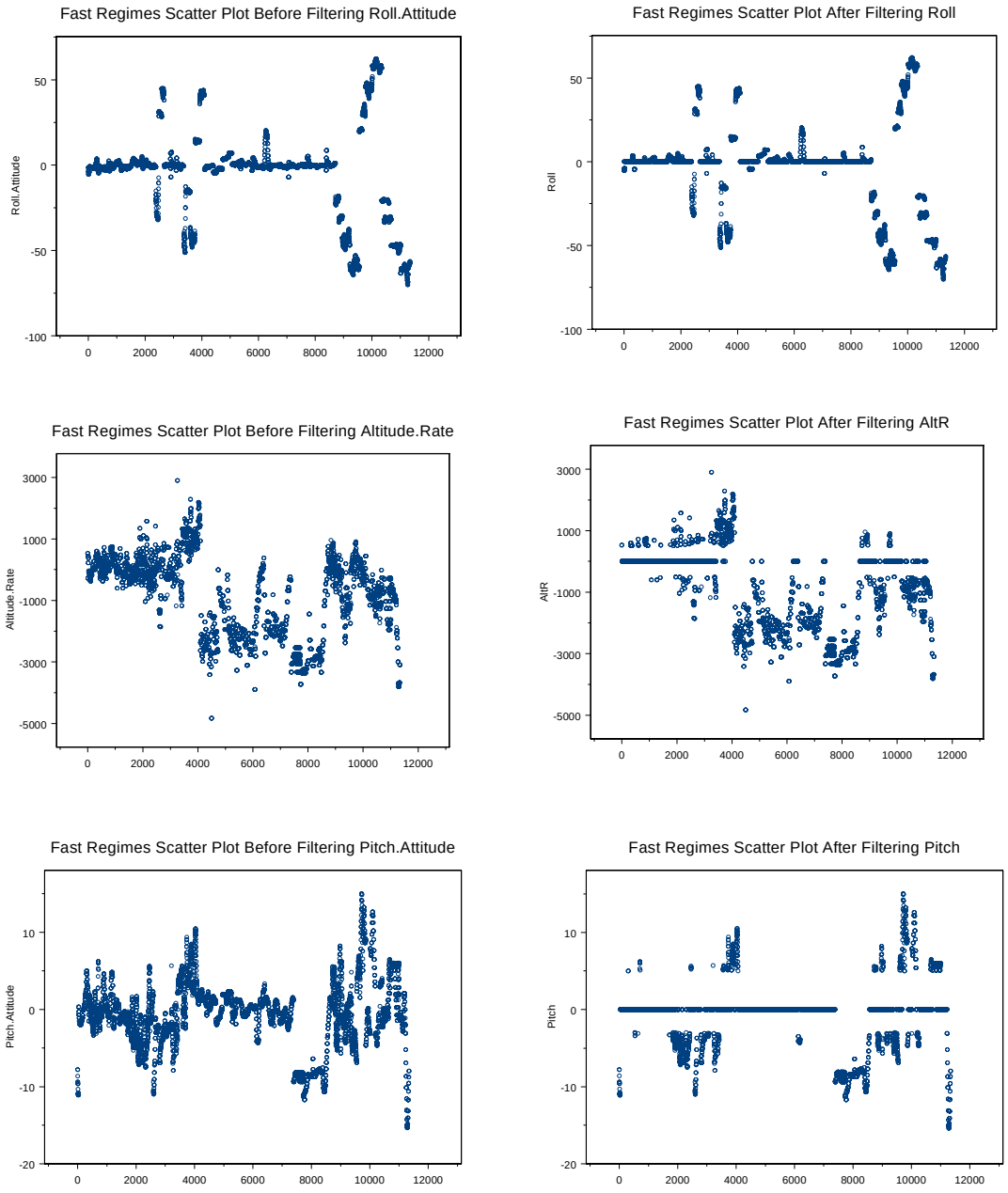


Figure 30. The Scatter Plots for Slow Regimes Parameter Values Before and After the Data Editing Process

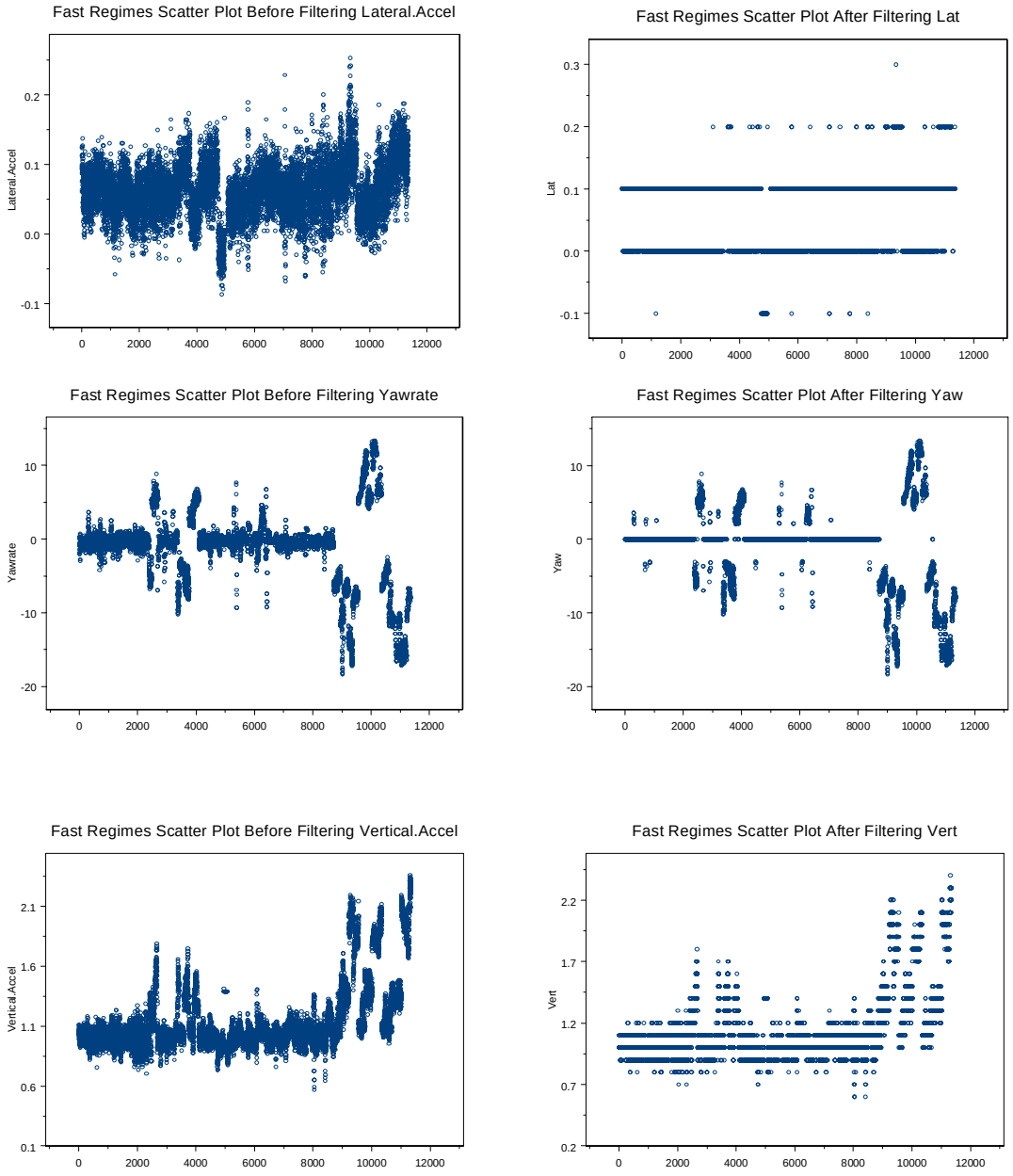


Figure 31. The Scatter Plots for Fast Regimes Parameter Values Before and After the Data Editing Process

THIS PAGE INTENTIONALLY LEFT BLANK

IV. METHODOLOGY and MODEL FITTING

A. METHODOLOGY

The goal of this analysis is to build a classification tree model that uses input parameters to predict the regime that the aircraft is in. The best model should contain the minimum number of bad misclassifications while predicting the regime. Along with high correct classification rates, the model outcomes should be interpretable, that is “make sense.” The model should produce a set of decision rules that can be evaluated by comparing them to the physical rules. Different types of tree-based methods are applied to the training set in order to have a sample space of tree models from which the reasonable model could be selected. After that, all the models in that space are analyzed using the test set to inspect their correct classification rates. Some models, even if they have a high correct classification rate, did not have a reasonable interpretation or they simply could not be validated. For validation, a very simple approach was used, that the order of parameters to split on. For example, to classify a right turn regime, the algorithm, the ruleset should have binary split on an unimportant parameter, such as *Nr*. The desired order of parameters is using *Weight.On.Wheels, KCAS, Roll.Attitude and Altitude.Rate.* After selecting the algorithm which produces superior results, a detailed study was carried out to construct a better tree using that algorithm to reach the best model. The following section gives the definitions and important features of the classification algorithms which were applied.

1. Tree-Building Methods and Algorithms

A classification tree is an empirical rule for predicting the class of an object from values of predictor variables (SPSS Whitepaper, 1999). Different tree algorithms all carry out basically the same steps; they examine all of the fields of the database to find the one that gives the best classification or prediction by splitting the data into subgroups. The process is applied recursively, splitting subgroups into smaller and smaller units until the tree is finished (as defined by certain “stopping criteria”). The target and input fields used in tree building can be

numeric ranges (continuous) or categorical, depending on the algorithm used. If a range target is used, a regression tree is generated; if a categorical target is used, a classification tree is generated (Clementine 10.0 Software Reference Notes). One of the main attractions of a classification tree is its simplicity: it performs binary splits on single variables in a recursive manner. Classifying a sample may require only a few simple tests. Yet despite its simplicity, it is able to give performance superior to many traditional methods on complex nonlinear data sets of many variables (Webb, 2002).

A node is a test on a parameter value. A branch represents an outcome of that test. A leaf (terminal) node represents a response or class. At each node, one parameter is chosen to split training examples into distinct responses/classes as much as possible. A new case is classified by following a matching path to a leaf node. Tree construction is a top-down process: in the beginning, all training examples are at the root. Then by choosing one attribute each time, the examples are partitioned recursively. Tree pruning is a bottom-up tree process; it removes sub-trees or branches, in a bottom-up manner, in order to improve the estimated accuracy on new observations. The splitting attribute is chosen from the available with the object being to improve or a “goodness score”. A goodness (purity) function is used for this purpose. Typical goodness functions include information gain (as in ID3/C4.5/C5.0 algorithms), and the information gain ratio Gini index (Lanzi, 2003).

In the classification and regression tree approach, six general questions arise:

1. How many decision outcomes or splits will there be at a node?
2. Which property should be tested at a node?
3. When should a node be declared a leaf?
4. If the tree becomes too large, how can it be made smaller and simpler?
5. If a leaf node is impure, how should the category label be assigned?

6. How should missing data be handled? (Duda, Hart, &Stork,1997).

Different combinations of the answers to the six questions above lead to different methods and algorithms.

a. The Classification and Regression Trees

This method uses recursive partitioning to split the training records into segments with similar output field values. The predicted response for all observations in a leaf is the level with the largest probability in that leaf.

(1) Measures of node impurity. The impurity of a node is “0” if all of the patterns that reach the node bear the same category label, and is large if the categories are equally represented.

The most popular measure is the entropy impurity (occasionally called information impurity). $P(w_j)$ is the fraction of training patterns x at node N that are in category j, given that they have survived all the previous decisions that led to the node N. Then

$$\text{EntropyImpurity(N)}=-\sum P(w_j)\log P(w_j)$$

Another measure is Gini impurity. This a generalization of the variance of a distribution associated with two or more categories.

$$\text{GiniImpurity(N)}=\sum_{i\neq j} P(w_i)P(w_j)=1-\sum_j P^2(w_j)$$

The misclassification impurity measures the minimum probability that a training pattern would be misclassified at N.

$$\text{MisclassImpurity(N)}=1-\max_j P(w_j)$$

In multiclass binary tree creation, the twoing criterion may be useful. The overall criterion goal is to select the split that best separates groups of the *c* categories, i.e., a candidate super-category C1 consisting of all patterns

in some subset of the categories, and candidate super-category C2 as all remaining patterns. The twoing criterion is not a true impurity measure.

(2) Stopping criteria for splitting. One traditional approach is to use techniques of a particular cross-validation. That is, the tree is trained using a subset of the data (for instance 90%), with the remaining (10%) kept as a validation (test) set. One continues splitting nodes in successive layers until the error on the validation data is minimized. Another method is to set a (small) threshold value in the reduction in impurity; splitting is stopped if the best candidate split at a node reduces the impurity by less than that pre-set amount**.** This method has two main benefits. First, unlike cross-validation, the tree is trained directly using all of the training data. Second, the leaf nodes can lie at different levels of the tree, which is desirable whenever the complexity of the data varies throughout the range of input (Duda et al., 1997.)

(3) Priors, loss and weights. If a category *i* is represented with the same frequency in both the training and the test data, it will not affect the tree creation. If this is not the case, priors should be used as a method for controlling tree creation so as to have lower error on the actual final classification task when the frequencies will be different. The most direct method is to weight samples to correct for the prior frequencies as well as seek to minimize a general cost, rather than a strict misclassification or 0-1 cost**.** Such information can be presented in a cost matrix *C*. $c_{ij} \in C$ is the cost of classifying a pattern as class *i* when it is actually class *j*. Cost information is easily incorporated into a Gini impurity, using the following weighted Gini impurity, which should be used during

$$\text{WeightedGiniImpurity}(N) = \sum_{ij} c_{ij} P(w_i)(w_j)$$

training. Costs can be incorporated into other impurity measures as well (Duda et al., 1997.)

(4) Pruning. The goal of pruning is to prevent overfitting to noise in the data. There are two strategies for “pruning”: postpruning, which

amounts to taking a fully-grown decision tree and discarding unreliable parts; and prepruning, in which the algorithm stops growing a branch when information becomes unreliable. (Lanzi, 2003) The cost-complexity pruning penalizes the largest trees. The penalty α is incorporated into the score function, so that each node will add α to the overall score. If the α chosen is bigger, the tree will be smaller. (Whitaker, 2006) Therefore, the algorithms avoid growing bigger trees so as not to be penalized for adding more nodes. Using α, the complexity (size) of the model can be controlled by pruning. A small value of α will produce a very large tree. It is possible to prune a large tree to have a valid "right-sized" (small) tree which can achieve the same correct classification rate on new data as the large tree. The α using which the tree should be pruned can be found inspecting the complexity parameter plot of the tree model. The following plot shows the cross-validated estimate of error in the y-axis and the complexity parameter in the x-axis.

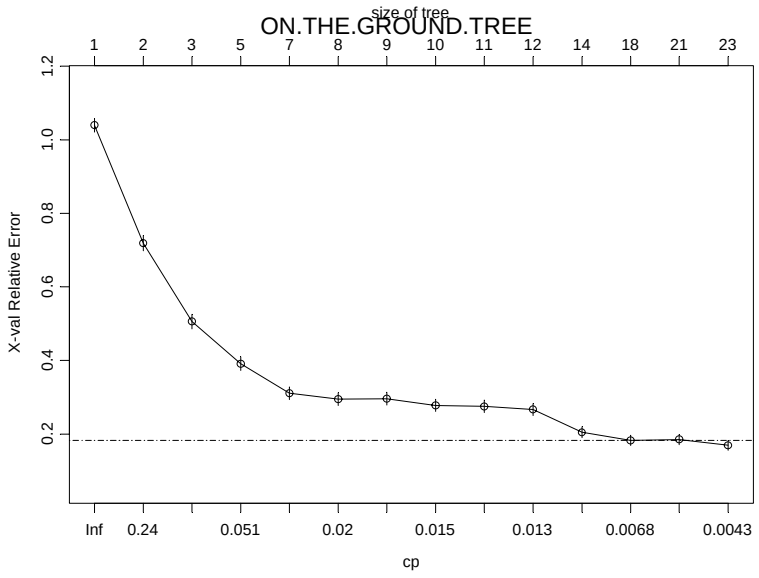


Figure 32. A Complexity Parameter Plot

The area where the curve gets flat, that is the error rate stops decreasing as quickly as before, gives a good complexity parameter value for the tree model. The tree should be pruned using this complexity value. After

pruning, the outputs tree models should be checked if they are able to classify as well as the pruned (larger) one. A smaller tree is easier to interpret and evaluate. A very common approach for pruning is to grow a large tree and prune it using Breiman's One Standard Error Rule (StatSoft, n.d.b)

(5) Handling missing values. Classification models might have missing attributes during training, during testing, or both. There are several different ways to handle this problem. How the algorithm handles the missing values was one of the criteria for the selection of the algorithm to establish the best model. Missing observations are often excluded when the model is training. When predicting an observation with missing data once it has fallen as far as it can the (non-terminal) node in which it lands gives the prediction. An alternative is the method of fractional cases. If 80% of X_1's go left, and X_1 is missing, then prediction = 0.8 * (left prediction) + 0.2 * (right prediction).The last alternative to apply is to use "surrogate splits," that is back-up splits computed when the tree is built (Whitaker, 2006.) The values of the parameter *Radar.Altitude* which were greater than 100 were set to "NA", because *Radar.Altitude* cannot be an input variable to classify the regime other than in-ground-effect regimes. Therefore, the algorithms that were selected to build classification models had the capability to use the missing values. Both Rpart and C5.0 algorithm have intelligent ways of handling missing values.

b. Chi-squared Automatic Interaction Detection (CHAID)

This method builds classification trees by using chi-square statistics to identify optimal splits; however, there are more areas wherein this method differs from classification and regression tree algorithms. The basic algorithm that is used to construct (non-binary) trees for classification problems (when the dependent variable is categorical in nature) relies on the Chi-square statistics to determine the best next split at each step. For regression-type problems (continuous dependent variable) the program will actually compute *F*-statistics. Specifically, the algorithm proceeds as follows:

(1) Preparing predictors. The first step is to create categorical predictors out of any continuous predictors by dividing the respective continuous distributions into a number of categories with an approximately equal number of observations. For categorical predictors, the categories (classes) are "naturally" defined; that is they are used the way they are held in the data.

(2) Merging categories. The next step is to cycle through the predictors to determine for each predictor the pair of (predictor) categories that are least significantly different, with respect to the dependent variable. For classification problems (where the dependent variable is categorical as well), the algorithm will compute a Pearson Chi-square statistic; for regression problems (where the dependent variable is continuous), it will compute an *F* statistic. If the respective test for a given pair of predictor categories is not statistically significant, as defined by an alpha-to-merge value, then it will merge the respective predictor categories and repeat this step (i.e., find the next pair of categories, which now may include previously merged categories). If the difference between response values for the pair of predictor categories is significant (less than the given alpha-to-merge value), then (optionally) it will compute a Bonferroni adjusted *p*-value for the set of categories for the respective predictor.

(3) Selecting the split variable. The next step is to choose the split predictor variable with the smallest adjusted *p*-value, i.e., the predictor variable that will yield the most significant split. If the smallest (Bonferroni) adjusted *p*-value for any predictor is greater than some alpha-to-split value, then no further splits will be performed, and the respective node is a terminal node (StatSoft, n.d.a)

c. Quick, Unbiased, Efficient, Statistical Tree (QUEST)

Classification trees based on exhaustive search algorithms tend to be biased towards selecting variables that afford more splits. As a result, such trees should be interpreted with caution. However, QUEST has negligible bias (Loh & Shih, 1997) (as cited in SPSS Whitepaper,1999). QUEST is also a tree-structured classification algorithm that yields a binary decision tree like C&RT. The reason for yielding a binary tree is that a binary tree allows for techniques such as pruning, direct stopping rules and surrogate splits to be used. Unlike CHAID and C&RT, which handle variable selection and split point selection simultaneously during the tree growing process, QUEST deals with them separately. It is well known that exhaustive search methods such as C&RT tend to select variables with more discrete values, which can afford more splits in the tree growing process. This introduces bias into the model, which reduces the generalizability of results. Another limitation of C&RT is the computational investment in searching for splits. The QUEST method is designed to address these problems. QUEST has been demonstrated to be superior to exhaustive search methods in terms of variable selection bias and computational cost. In terms of classification accuracy, variability of split points and tree size, however, there is still no clear winner when univariate splits are used. The QUEST algorithm for each split, the association between each predictor variable, and the target are computed using the ANOVA *F*-test or Levene's test (SPSS Whitepaper,1999). However, this algorithm is very slow and impractical for big data sets such as the regime recognition data. When this algorithm is applied to the training data set, the computer system runs out of dynamic memory.

d. C5.0 Tree-Building Algorithm

A C5.0 model is an algorithm that works by splitting the sample based on the field that provides the maximum information gain. Each sub-sample defined by the first split is then split again, usually based on a different field, and the process repeats until the sub-samples cannot be split any further. Finally, the lowest-level splits are reexamined, and those that do not contribute significantly to the value of the model are removed or pruned. C5.0 requires a categorical response to fit a tree model (Clementine 10.0 Software Reference Notes). This decision tree algorithm is the unpublished commercial version of C4.8. A detailed approach of this algorithm follows:

1. Choose an attribute that best differentiates the output attribute values.
2. Create a separate tree branch for each value of the chosen attribute.
3. Divide the instances into subgroups so as to reflect the attribute values of the chosen node.
4. For each subgroup, terminate the attribute selection process if:

    - All members of a subgroup have the same value for the output attribute, terminate the attribute selection process for the current path and label the branch on the current path with the specified value.
    - The subgroup contains a single node or no further distinguishing attributes can be determined. As in (a), label the branch with the output value seen by the majority of remaining instances.

5. For each subgroup created in step that has not been labeled as terminal, repeat the above process (Kdnuggets, 2006, March 11.)

C5.0 can produce two kinds of models. A decision tree is a straightforward description of the splits found by the algorithm. Each terminal (or "leaf") node describes a particular subset of the training data, and each case in

the training data belongs to exactly one terminal node in the tree. In other words, exactly one prediction is possible for any particular data record presented to a decision tree. In contrast, a ruleset is a set of rules that tries to make predictions for individual records. Rulesets are derived from decision trees and, in a way, represent a simplified or distilled version of the information found in the decision tree. Rulesets can often retain most of the important information from a full decision tree but through a less complex model. However, rulesets do not have the same properties as decision trees. The most important difference is that with a ruleset, more than one rule may apply for any particular record, or no rules at all may apply. If multiple rules apply, each rule gets a weighted "vote" based on the confidence associated with that rule, and the final prediction is decided by combining the weighted votes of all of the rules that apply to the record in question. If no rule applies, a default prediction is assigned to the record. The ruleset presentation is useful if it is desirable to see how particular groups of items relate to a specific conclusion. For example, the following rule offers a profile for a group of cars that is worth buying (Clementine 10.0 Software Reference Notes):

IF engine_in_good_condition = 'yes' AND mileage = 'low' THEN ‘BUY’

Some data mining software packages enable users to use boosting, cross-validation, and pruning to define an expected noise in the data. The next section describes some of the options available in Clementine.

(1) Boosting. Boosting is a process by which a number of trees are grown and their predictions combined in the final model. Boosting involves fitting a tree with equal weights on all observations in the training set, $T_1(x)$ and estimating training error (γ_1) as a function of the error rate of this tree. This measures how much better the model is than a naïve model. For example, on binary problems, estimating training error (γ_1) will measure $1/2 - ErrorRate$ (large error gives smaller γ_1.) Then the algorithm re-weights the observations; larger weights will be given to those observations which are misclassified. After

the first procedure, it fits a new tree $T_2(x)$ .and estimates γ_2. Then a third model is built to focus on the second model's errors, and so on. The final vote is a weighted vote or a weighted average of estimated probabilities (Whitaker, 2006.) Finally, the cases are classified by applying the whole set of models to them, using a weighted voting procedure to combine the separate predictions into one overall prediction. Boosting can significantly improve the accuracy of a C5.0 model, but it also requires longer training. The Number of Trials option in Clementine allows the user to control how many models are used for the boosted model (Clementine 10.0 Software Reference Notes.)

(2) Pruning. To prevent overfitting effects of the boosting, the pruning process is necessary in order to carry out this algorithm.

(3) Expected noise level. The expected proportion of noisy or erroneous data in the training set. If the training and the test data have different noise levels, a problem in the prediction may result.

(4) Automatic cross-validation. C5.0 will use a set of models built on subsets of the training data to estimate the accuracy of a model built on the full data set. This is useful if the data set is too small to split into traditional training and testing sets. The cross-validation models are discarded after the accuracy estimate is calculated. The number of models used for cross-validation can be specified (Clementine 10.0 Software Reference Notes).

e. Recursive Partitioning and Regression Trees (Rpart)

This decision tree algorithm differs from the set of routines that fit classification and regression trees in the areas stated below:

(1) Choice of splitting criterion. For regression trees, the default is that Rpart splits only by minimizing the sum of the two child RSS's. For classification trees, though, one can choose Gini or information splitting. Rpart will also produce trees in which the underlying response variable is assumed to be Poisson, or where it is a survival object using exponential lifetimes.

(2) Automatic cross-validation. By default, Rpart runs ten cross-validations and stores the results. This makes it easy to prune the tree. The number of cross-validations can be defined by the user (Therneau & Atkinson, 2000.)

(3) Ability to include loss matrix and/or prior probabilities.

(4) Weights for classification trees.

(5) Surrogate and competitor splits: Rpart finds five surrogate splits and four competitor splits. “Competitors” just indicate the second-best, third-best and so on split at each node; however, this might be useful to the analyst.

(6) Intelligent NA handling: By default, Rpart uses an intelligent missing value handling scheme in which the missing observations are essentially ignored split-by-split, while the usual algorithms omit observations with any missing values all the way through the tree-building process (Whitaker, 2006).

2. Other Classification Models

a. Logistic Regression

Binomial (or binary) logistic regression is a form of regression which is used when the dependent variable is a dichotomy and the independent variables are of any type. Multinomial logistic regression exists to handle the case of dependents with more classes than two. When multiple classes of the dependent variable can be ranked, then ordinal logistic regression is preferred to multinomial logistic regression. Continuous variables are not used as dependents in logistic regression (Garson, 1998.)

Logistic regression can be used to predict a dependent variable on the basis of continuous and/or categorical independents and to determine the percent of variance in the dependent variable explained by the independents; to rank the relative importance of independents; to assess interaction effects; and to understand the impact of covariate control variables (Garson, 1998).

Logistic regression applies maximum likelihood estimation after transforming the dependent into a logit variable (the natural log of the odds of the

dependent variable taking the value “1”.) In this way, logistic regression estimates the probability of a certain event occurring. Note that logistic regression calculates changes in the log odds of the dependent, not changes in the dependent itself as OLS regression does (Garson, 1998.)

b. Neural Networks

Neural networks are a class of models inspired by biological neurons. In the data mining world, they are used for various modeling problems such as prediction, classification and clustering. Neural networks are organized in layers: Input, hidden, and output. Each layer is a collection of artificial neurons. The neurons in one layer are connected to neurons in the next layer. The connections have weights. Fitting a neural network model is finding the values of these weights. Weights are found by Feed forward Back propagation algorithm, which is a form of Gradient Descent Method. Network architecture as well as certain training parameters is decided upon by trial and error. One should try various choices and choose the one that gives lowest prediction error (Saha, n.d..)

B. MODEL FITTING

In this section, the models and the algorithms defined in the previous section will be applied to the training dataset and the best one will be chosen using the test and validation set. S-Plus was used for data editing and fitting recursive partitioning and regression trees. The Clementine data mining system was used for the other models. This software has the ability to generate rulesets. The advantages of these are used in interpreting and validating the best model chosen.

At first, the algorithms are directly fitted on the training set. After inspecting the produced outcomes, it was understood that further modeling processes were required to build better models (See Data Editing for Model Fitting in Chapter III.)

After generating a variety of trees using algorithms presented in a variety of software packages, the next step is to eliminate the models with lower correct classification rates. The rulesets are a set of “if then” statements from the splits. These rulesets were tested for validity by checking the logical statements whether or not they really lead to the expected regime. During this process, the logical statements are also inspected to see whether or not they comply with the physical flight rules. For example, a ruleset which used only engine temperature, angle of bank and velocity to classify a descending right turn did not pass this step, because there are more important parameters which give better (or more accurate) information on that regime such as altitude rate. The best algorithm was chosen by deciding which algorithm produced a model that classified a given regime using the important parameters for that regime. This criterion was also used for checking the validity of the sub-models.

When the best algorithm was chosen to build the final model, several problems were encountered. Those problems were given in section “Remodeling with C5.0 to Fix Problems.” The problems are solved by muting (See Chapter III, Data Editing Process for Model Fitting), subsetting and only using relevant parameters as input.

The advantages that the software packages offer to users include many powerful abilities such as boosting, cross-validating, pruning and defining an approximate level of noise in the data. These are all used in every step of fitting. The next section offers information about the methods fit to the data (see Figure 33). Another model is fit using the Rpart library in SPlus. The same procedures are applied. For preliminary division in this model, not all of the parameters used for preliminary division process in the C5.0 model were used (see Figures 36, 37, 38, and 43.) Rpart was used to build the classification model (see Figure 44.)

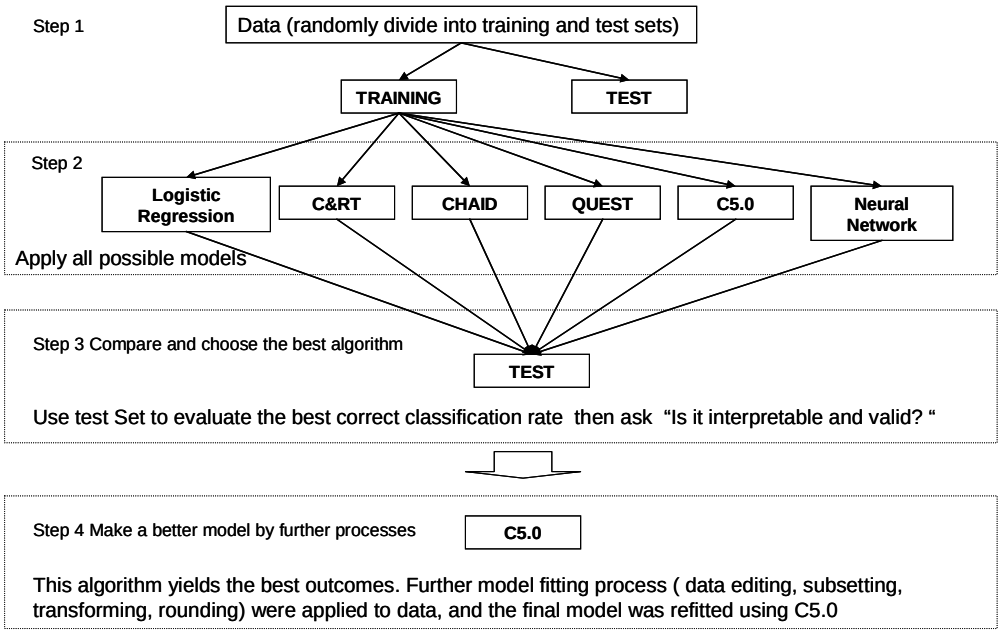


Figure 33. The modeling process with C5.0 (Clementine)

1. The Classification and Regression Tree (C&RT)

This algorithm was applied to the training set and the model produced was analyzed using the test set. The Clementine data mining system was used. The correct classification rate for this classification was about 50%. In the coincidence matrix, it is very obvious that some regimes are never predicted. The subsets of similar regimes were called "families" of regimes. These regimes' numbers are in a sequential order: for example, Regime 21 is a level flight between 0.5-0.6 $V_{h,}$ while regime 22 is a level flight between 0.6-0.7 V_h (see Table 3 in Chapter III.) It would be acceptable if regime 22 were classified as regime 21; it would not be a bad misclassification. On the other hand, if a model classified an autorotation regime as a level flight, that would be a bad classification. (In an autorotation regime, the aircraft is losing altitude and torque is at a very low level, but in a level flight regime, the aircraft is not climbing or descending.) In the coincidence matrix, most misclassifications happen to be to one of the neighborhood regimes. The level flight misclassifications are dispersed into level flight regimes. The same rule is also applicable to the hover regimes. Some regimes of the hover family which are not observed in the set of the predicted regimes were classified

as other hover regimes. However, most of the autorotation regimes were found in the level flight regimes. These misclassifications are bad ones. Apparently, the best classification rates are observed for the regimes that consist of a banked turn. The underlying reason for these high rates may be the definite pattern in the parameter roll attitude. The change in this parameter from a level flight to a banked turn is easily captured so that the algorithm chooses this parameter as a very interesting parameter on which to split. The same approach is also valid for high-speed regimes. The algorithm may also find the speed as an interesting one on which to split. On the other hand, at low speeds, the dynamic environment around the aircraft affects the pitot system. Therefore, there can be unexpected readings for speed which result in unexpected values. It is easier to capture regime changes at higher speeds. For the on-the-ground regime family, most of the time, the misclassifications happen to disperse into other on-the-ground regimes. The only important problem in this regime family is that aircraft take-offs are often misclassified. In some cases, a number of them are classified as level flights**,** which are very bad misclassifications. In the coincidence matrix, there is also some inconsistent behavior. Most of the regimes are classified as level flights. Even though this model gives a very low correct classification rate, it is still applicable. A better model would be possible by rearranging and collapsing some levels of regime. This may be done by partitioning the regimes into family (neighborhood) groups and giving a single level of regime number to all of that family's members. It is assumed that doing so would increase the correct classification rate a great deal, but this idea was reserved. In any case, C5.0 gives better outcomes than C&RT; so this algorithm will not be used to build the final model.

In the algorithm, there is also a cost matrix which is incorporated into the impurity measurement. By default, this matrix has "1" everywhere except on the diagonal where the values are "0." This means classifying *Regime i* as *Regime i* has no penalty in effect, but classifying Regime *i* as *Regime j* has a penalty of "1." This cost matrix can be redefined to incorporate the user's penalty

preference. In the cost matrix, a larger cost was assigned to a number of bad misclassifications. The expectation was that these bad misclassifications would be corrected. However, the bad misclassifications which were penalized became different bad misclassifications. The correct classification rate was improved by about only 10%. If there were a smaller number of regimes, using a penalty matrix may give better results, since the sample space for classifications (and bad classifications) would get smaller (see Appendix A for the coincidence matrix.) Since the C5.0 algorithm yields superior results, the approach of using a cost matrix in C&RT in Clementine was not used.

2. Chi-squared Automatic Interaction Detection (CHAID)

This model was fit on the training set and applied to the test set using Clementine. The correct classification rate for this model was about 80%. By applying this algorithm, a larger correct classification rate was observed than with C&RT. In the coincidence matrix, most of the misclassifications clustered around the diagonal which suggests that the misclassifications are not very bad. However, the correctness of the classification cannot be directly understood through observation of the closeness to the diagonal. For example, regime 5 and regime 51 belong to totally different regime families, but in the coincidence matrix they are listed one after the other (see Table 3 Chapter III.)

With the previous algorithm, there were many problems with the level flight and hover regimes. Here these problems are reduced to a lower level. However there are a small number of bad classifications for the hover turns. These regimes are classified as banked turn regimes. All of the flights which are right slips in autorotation were classified as descending banked turns.

In the previous model and in this model, low speed and on-the-ground regimes tend to be misclassified. Especially with hover and on-the-ground regimes, it may be difficult for the algorithm to capture small changes in the parameter values, but those small changes change the regime of the aircraft. Take-off regimes were classified as taxi, or vice versa. Hover sideward flight regimes happened to be classified as hover turns, and so on.

This model would make a lot of sense if the regimes were collapsed into families. CHAID is more capable of finding the optimal splits**,** which allows for a smaller number of bad classifications than the previous model. (see Appendix A for the coincidence matrix). Since the C5.0 algorithm yields superior results, this algorithm was not used.

3. C5.0 Tree-Building Algorithm

This algorithm was applied to the training set and analysis to the test set using Clementine. This model has the largest correct classification rate, around 97%. Just as in the previous model, there are misclassifications observed for low speed regimes. The most definite one is the misclassification of taxi regimes. Most of the time those regimes were classified as take-off regimes.

For correctness purposes only, this model is stronger than the previous one. This statement is also true for reasonable misclassifications. By using this model, the collapsing of the levels of the regimes is no longer needed. The only big problem is the possible overfitting in the model. To prevent it, the expert options were used to severely prune the tree and stop splitting after attaining a threshold of information gain.

The model was validated by checking the generated tree to discern whether or not it makes sense. Since there are about 50 regimes, the tree generated by the model can not be checked easily. Clementine has a valuable feature for its tree models, which is the ability to generate rulesets from the constructed tree. When these rulesets were created and analysis began, it was understood that there were a lot of redundant splits. The most definite ones are splits on small values of angle of bank, but since roll attitude and angle of bank give the same information. One of these parameters that contain the same information may not be needed in the model. *Radar.Altitude or PA* (pressure altitude) splits were not important splits for classifying level flights. *KIAS* and *KCAS* splits were unnecessary in the presence of the *Vh.fraction* in the model. Some parameters, such as *Nr*, never showed up in the splits of the rulesets. Since it is just a percentage value and does not change very much accross

different regimes, it is not an interesting one to split on. Another parameter that did not show up in the rulesets was *Take.off.Flag*. Actually this parameter was a very important one, because if it is "0," it means the regime is a take-off regime regardless of the other parameters. The same argument is also true for *Weight.on.Wheels. Weight.on.Wheels* should be the first parameter to split on, but the logical statements (splits) on this parameter were observed in (lower) inner layers of ruleset. To make sure that the C5.0 algorithm used *Take.off.Flag* and *Weight.on.Wheels,* these two parameters were used to divide the full data into smaller datasets (see Figure 36.)

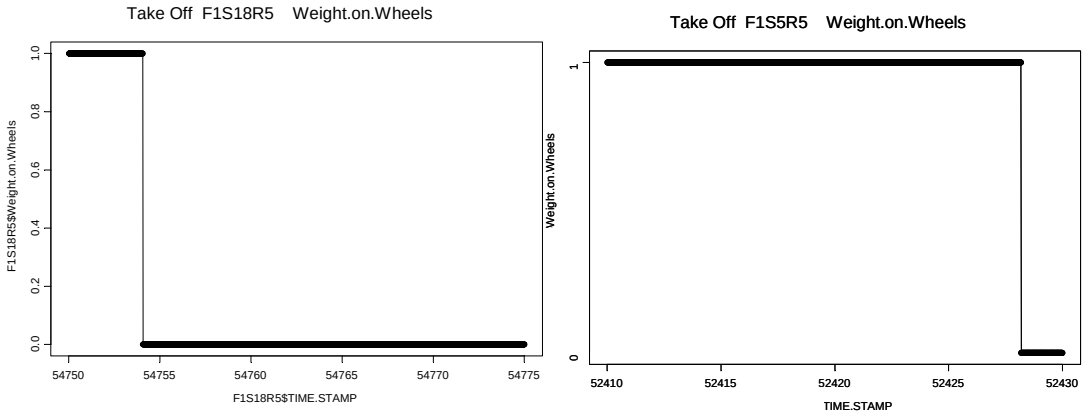


Figure 34. The Behavior of The Parameter Weight.On.Wheels

In the plot above left, the x-axis shows the time frame for a take-off regime and the y-axis shows the values for *Weight.on.Wheels.* Only a small proportion of the observations has the value of "0." In contrast, in the plot above right, the parameter has the value of "1" most of the time. The plots above are from two different take-offs which produced two different patterns for *Weight.on.Wheels.* In the first pattern only a small proportion of the observations was "1" and in the second one only a small proportion was "0." For the same regime, the parameter values did not present the same information. These two different patterns are not consistent. As expected, the value changes for *Weight.on.Wheels* were observed only in take-off regimes. Therefore, there were only two sets of observations for the take-off regime that could be used in classification process. The algorithms

had to use this inconsistent information on *Weight.on.Wheels,* and this prevents *Weight.on.Wheels* from being the first parameter to split on. If the values of *Weight.on.Wheels* had the same pattern and overlapped most of the time, then *Weight.on.Wheels* would supply consistent information to the algorithms and become the first split. Since some planned (expected) values were not observed for *Control.Reversal.ID*, there were no splits on this parameters or the splits are in the undesired layer.

The problems encountered in model validation are summarized below:

1. Unnecessary splits on the small values of some parameters
2. Very important categorical parameters (such Weight.on.Wheels) are not in the rulesets or they are at lower (inner) layers of the rulesets, which means they are not in effect at the right place.
3. Redundant parameters in the model.

For all of the reasons given above, this model, despite its high correct classification rate, cannot be accepted as valid.

4. Remodeling with C5.0 to Fix Problems

This section focuses on the procedures used to overcome the modeling problems discussed at the end of the previous section (see Appendix B.)

a. Unnecessary Splits on the Small Values of Some Parameters

To prevent this problem, values of some of the important parameters in a usual are muted by setting to a default value**,** i.e., “0.” For example, the aircraft will have a roll angle to either side at an level which is caused by the balancing forces or the environment. If the roll angles are within an acceptable interval, they can be set to “0.” This process made the algorithm think that those values are not interesting enough to split on, and may aid in clearing in clearing the noise in the rulesets. Eventually the ruleset will become more interpretable or reasonable (see Data Editing Process for Model Fitting, Chapter III.)

b. Very Important Categorical Parameters are Not in the Rulesets

Categorical parameters which predominantly take one level are not often used for splitting, even though they might be very important for classification. One solution is to build sub-models by filtering the training data using different parameters, and fitting trees to those smaller data sets. This method will ensure that the parameter changes or the different levels are captured by the model.

(1) By partitioning the training set into subsets. The full data set will be divided into smaller data sets. The parameters used for this are *Weight.on.Wheels, KCAS, and Control.Reversal.ID.* Filtering the data using *Weight.on.Wheels* will make two smaller data sets. One of them will have “0” for all *Weight.on.Wheels* parameter values. The other one will have “1.” The smaller data set which has “0” for *Weight.on.Wheels* will be called *In-the-air* data set. The other one will be the *On-the-ground* data set. The resulting in-the-air data set will be filtered into fast and slow (low speed) regimes. The cut-off value for the speed, 43 knots, which was used in the Goodrich documents as a threshold value, was observed in some rulesets as a primary split. To have a minimum number of unwanted regimes in subsets, 44.72 knots will be used as the cut off value. This value was determined by trying out some values that are found by visual inspection of the plots. The value which results in the smallest number of unwanted regimes in different families is 44.72. Finally, those small data subsets of the *In-the-air* data set will be divided into two by using the presence or absence of Control.Reversal.ID. A further division and filtering is also applicable for *Landing.Flag* and *Take.off.Flag* (see Figures 36, 37, 38.) The same subsetting in Figure 38 was also applied to the “In the air and fast” data set.

(2) Fitting a model to each subset. Now the subsets have regimes that belong to the same regime family. Even if the model wrongly classifies a regime, the misclassification will be in that family. On the other hand,

filtering also causes some regimes to appear in more than one subset. The most readily apparent one is the take-off regime. The *Weight.On.Wheels* parameter values for Take-off is “1” as long as the aircraft’s altitude above ground level is “0”; after the aircraft is off the ground**,** this parameter turns into “0.” This problem also arises from instantaneous and intended changes in the parameters which were used for subsetting. An instantaneous drop in the speed in a level flight, even if the true value if the regime belongs to the high speed family, will cause these observations to go to the low speed subset. They will, however, be in the wrong family and the model will include those regime numbers into the model fitting and use them for predictions.

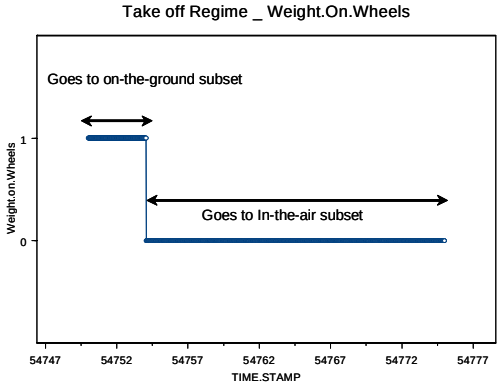


Figure 35. The Behavior of a Take-off Regime in Subsetting Process

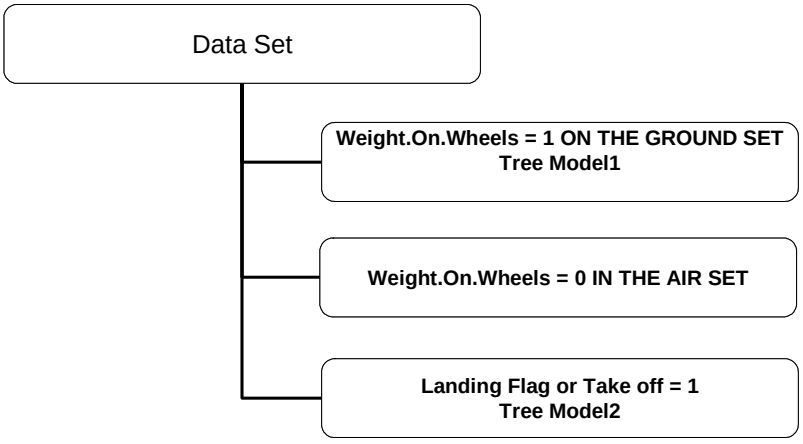


Figure 36. Subsetting the Big Data into Smaller Sets (WOW, Flags)

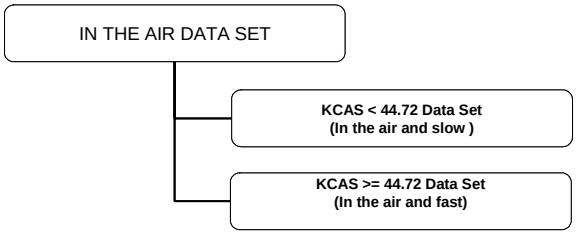


Figure 37. Subsetting “In the air data“ Using *KCAS*

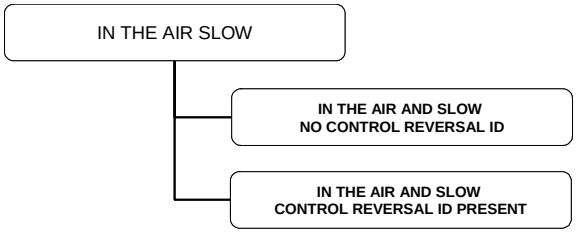


Figure 38. Subsetting “In the air and slow” Dataset Using *Control.Reversal.ID*

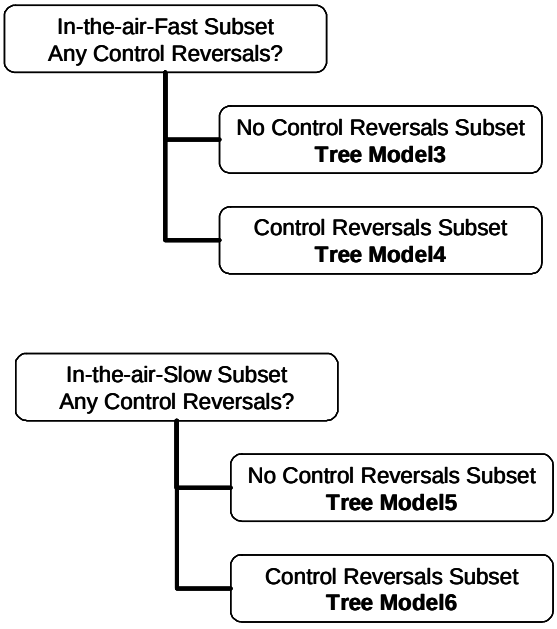


Figure 39. The Names of the Sub-trees

(3) Applying the sub-models to the training and test data. The training data is divided into subsets using filters**;** at each terminal node a tree

model is fitted**,** and a ruleset is generated. The same filtering will be applied to the test set; then the subset is directed to the corresponding rulesets. Finally, there is a coincidence matrix for analysis of each model.

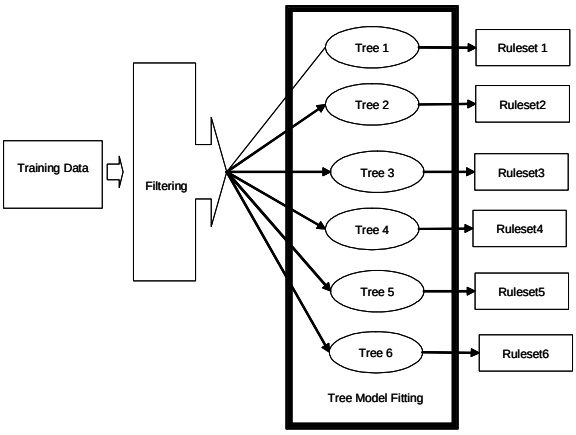


Figure 40. The Filtering and Model Fitting Stream (see Appendix B.)

(3) Analyzing the rulesets and Validation. Each ruleset is checked to discover if it really makes sense. This was done by asking two different pilots. The pilots were given the “if” statements” and asked to name the approximate regime. Most of the time, they were able to correctly classify the regime using the if-statements of the rulesets. In a sense this process can be called model validation with a practical approach. The only potential difficulty with the model was that there were still some unnecessary parameters that could be taken out of the model. The next section discusses this issue.

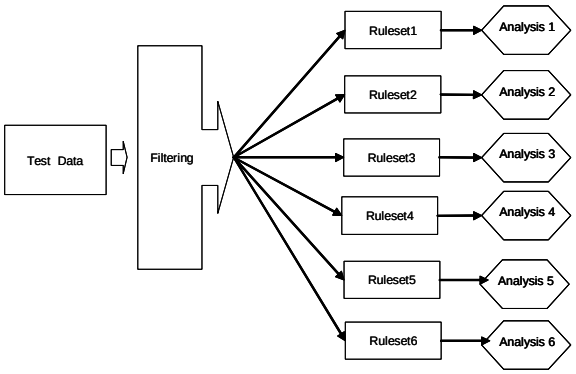


Figure 41. The Filtering and Testing Stream

c. Redundant parameters in the model

Parameters which were unnecessary were taken out of the model. For example, *TGT* is unnecessary, because it gives information about the engine limits. We do not need that information since we do not have such a regime present in the dataset. *Weight.On.Wheels* and *KCAS* are already out of the model since they were used to filter the data. *Vertical.Accel* and *RateOfClimb* (*RC*) are not needed in the model since *Altitude.Rate* gives nearly the same information. *Radar.Altitude* gives some information on the regime up to the out-of-ground-effect hover altitude. The presence of values bigger than this hover altitude in the model may cause numerous misclassifications because an aircraft can be in the same regime at different altitudes (or vice versa). Therefore, only the values up to the out-of-ground-effect hover altitude were included in the model by setting the larger values to NA. Finally, *Time.Stamp* should never be used.

5. Recursive Partitioning and Regression Trees (Rpart)

For this model, the data sets are subset using only *Weight.On.Wheels* and *KCAS.* There are three different sub-tree models: one is for the “On the ground” data set, one is for the “In the air and slow,” and the last one for the “in the air and fast.” Not all of the parameters are used as input parameters; only the ones most likely to give the best information on predicting the regime are used. For the “On the ground“ model, no data editing process was applied. None was needed because as Table 4 indicates, the number of the regimes for this family is not numerous.

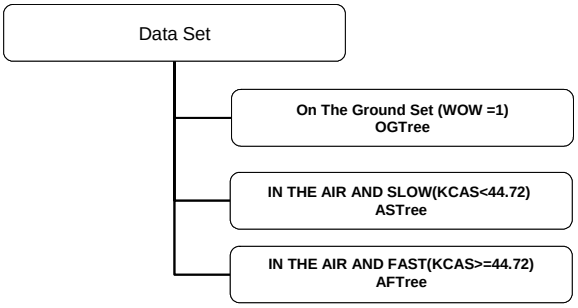


Figure 42. The Data subsets for Rpart Model

The classification trees are large as a result of the large number of regime levels and the large number of parameters on which the algorithm split. The trees built are pruned as much as possible. At first, using a small complexity parameter, such as *cp=0.001*, a big tree is constructed. As expected this tree is very complex. The tree is then pruned applying Breiman's One SE rule. The rule is to simply find the cp (complexity parameter) for which (the cross validated estimate of error) xerror < (best xerror + best xerror's corresponding xstd). Actually, in this specific case, the tree can be pruned by a little more than the One SE rule with little resulting effect on the tree. The pruning process for a model with a 50-level-response is very sensitive, because some values of *cp*s do not produce a model that predicts all the levels of response. The number of predicted levels of the regime begins to get smaller than its actual value. The ones that are misclassified are often very bad classifications, and they appeared to be all around the coincidence (confusion) matrix. The One SE rule used to prune the trees may cause more complexity and size than trees produced by using a *cp* by visual inspection of cp plots. On the other hand, the One SE rule is better for our prediction purposes.

There are two main problems in modeling by subsetting on some parameters before fitting trees. The first one is that some observations go into the wrong subset. For example, the regime 5 is the take-off regime, but not all the observations in this regime have the *Weight.on.Wheels* parameter "0" or vice versa (see Figure 43.)

Since the problem stated above is caused by the nature of the data it can not be fixed using different data editing processes. Even if they are not in the correct families, they can be considered acceptable classifications.

The second problem caused by subsetting is a natural result of the first problem. Before partitioning, the distribution of the regimes is uniform, but in the smaller subsets it is not. To solve this problem, loss matrices which assign costs to misclassifications of different types are used. Different costs for

misclassification in CART can be modeled either by means of modifying the loss matrix or by means of using different prior probabilities for the classes, which again should have the same effect as using different weights for the response classes (Breiman et al. as cited in Williams, 2004.)

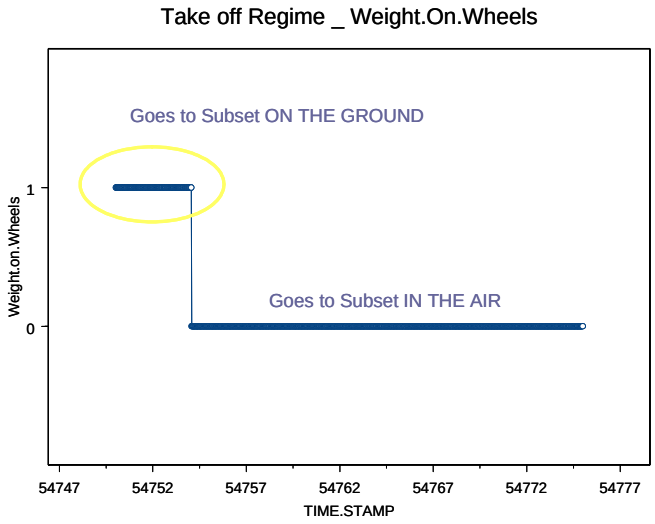


Figure 43. Weight.on.Wheels in a Take-off Regime

The loss matrices are created from the table of predicted values (coincidence matrix) of the test set. The tree is pruned using the One SE rule, and the predicted table is formed into a loss matrix by a function. For each element in the matrix, this function checks if that element is on the diagonal which means it is a correct classification and, assigns a penalty of zero to that element. If that element is not on the diagonal and greater than a threshold value, the function uses another function to determine its penalty depending on the absolute value of (i-j). If this value is small (i.e. close to the diagonal which means they are neighbors, and they belong to same family), a small penalty is assigned; if it is not, the greater penalty is assigned depending on the absolute value of (i-j). The threshold value used by the function is acquired by visual inspection of the coincidence matrix by finding a value less than which a lot of

misclassifications can be accepted as good misclassifications. For example, in the table below, 7 is selected as the threshold value. If the number of misclassification is greater than that value, it is considered bad misclassification. If the element is not on the diagonal and the value is less than the threshold value, it is considered a good misclassification and assigned a penalty of one (see Appendix E.)

	10	11	12	13	14	15	16	17	26	27	28	5	7	8	9
10	149	3	4	4	4	0	0	0	0	0	0	0	0	3	1
11	1	157	2	7	0	0	0	0	0	0	0	0	0	0	0
12	0	3	132	9	0	0	0	0	0	0	0	0	6	6	12
13	3	0	1	160	0	0	0	0	0	0	0	0	0	0	3
14	2	0	3	0	149	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	155	10	2	0	0	0	0	0	0	0
16	0	2	0	0	0	6	157	2	0	0	0	0	0	0	0
17	0	0	0	0	0	4	0	163	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	84	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	58	0	0	0	0	0
28	0	0	0	0	0	0	0	0	2	0	5	0	0	0	0
5	2	1	1	1	1	0	0	0	0	0	0	60	0	0	0
7	0	1	2	0	0	0	0	0	0	0	0	7	142	13	0
8	0	0	6	0	0	1	0	0	0	0	0	0	14	157	1
9	0	0	1	7	1	0	0	0	0	0	0	0	0	3	173

Table 10. Finding the Threshold Value for the Penalty Function

Using the loss matrix created in this way, a new tree model is fitted and pruned again. The test set is again directed into the model**,** and the coincidence matrix is formed. Since Rpart automatically divides the training set into subsets for cross validation, there is no large improvement in the outcomes. On the other hand, this approach guarantees that not using prior probabilities in the model is no longer an important issue (see Figure 44.)

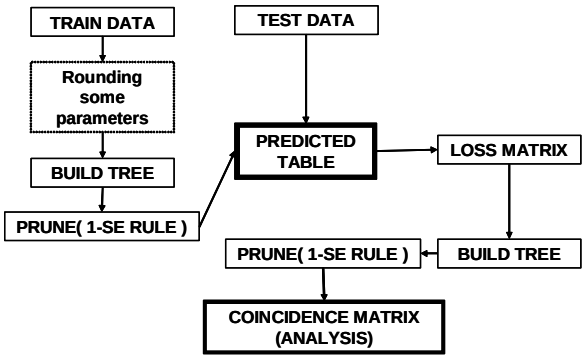


Figure 44. The Rpart Modeling Process

Even when the trees are pruned as much as possible, they are still very complex. The lower braches are not that important as long as the main branches are giving the correct sense of the classification. Therefore, to understand more clearly, the trees will be “snipped” (using the functions from the Rpart library) when printed. The Rpart model has quite reliable correct classification rates. The Rpart models give nearly the same results as the C5.0 models.

6. Other Possible Models

To make sure that the other options such as logistic regression or neural networks do not give better rates than the best model, these methods were applied.

a. Logistic Regression

This method was applied by using the first nine scores of the principal components of the continuous parameter and the categorical parameters as input variables. The first nine scores captured 93% of the variability in the data. The overall correct classification rate was about 55%.

b. Neural Networks

This model has a correct classification rate of about 55%. Actually, the algorithm performs better in sub-models where the number of levels of the response variable is smaller. Furthermore this model is more useful if the regimes are collapsed more into smaller families. Neural networks were not pursued farther in this study.

THIS PAGE INTENTIONALLY LEFT BLANK

V. RESULTS AND CONCLUSIONS

A. RESULTS

As mentioned in the previous chapter, C5.0 and Rpart models have superiority over the other possible models. These two models are selected as the best algorithms and further modeling procedures are used to obtain better models. In this chapter, these two models are analyzed and compared. In the table below a summary of the correct classification rates are given.

C5.0 Model	Correct Classification Rate	Rpart	Correct Classification Rate
On the Ground Model	87.6%	On the Ground Model	89.0%
Take-offs and Landings Model	85.3%	In the Air and Slow	92.6%
In the Air and Slow with CR	100.0%	In the Air and Fast	97.5%
In the Air and Slow w/o CR	92.2%		
In the Air and Fast with CR	100.0%		
In the Air and Fast w/o CR	99.5%		

Table 11. The Correct Classification Rates

The preliminary partitioning process minimized the number of bad classifications by fitting sub-models on each data subset. On the hand, this process caused some classification errors. There are two reasons: first is when unobserved values of the subsetting parameters, such as *Control.Reversal.ID*, caused those observations to be directed into wrong regime families, and the second is when instantaneous but unintended changes in the parameter values of *KCAS* directed those observations directed into wrong regime families. Those observations in the wrong families were included in modeling process in whichever the data subsets they fell into. The response parameters for those regimes were correctly predicted by the models, even if the observations were directed into the undesired families. In any particular flight, a single regime may seem to consist of several different regimes. For example, when the pilot tries to execute a right turn in a level flight regime, there may be some weather conditions which might prevent a smooth turn and the aircraft might have bigger values for *KCAS* just for a moment until this is corrected by pilot inputs. Therefore, a planned right turn in flight may actually contain not only a right turn regime but also some other regimes. Those other regimes may not even be

members of the same family. This is a problem for any model-fitting process since there is a response already assigned to those observations in that time interval. However, this is not a problem in normal operation, since those deviations from the plan really do represent different regimes being flown, even if only for a short period, and those different regimes should be recorded for usage monitoring. Observations of this sort -- representing isolated instances of one regime within a bigger set of observations of another regime -- will be directed into the wrong families but should not cause increased error rates in normal operation.

1. The Analysis and Results of the C5.0 Model

The C5.0 model was built using Clementine 10.0. The model outputs were rulesets. When reading the rules, if there are no if-statements on a parameter, it should be assumed that its values do not have interesting values; they are, therefore, in the intervals of usual parameter values. Having no “if” statements on the *Airspeed.Vh.Fraction*, for example, does not mean that the aircraft is not moving. Since the model consists of sub-models (sub-trees), those smaller models will be analyzed individually to reach an overall result.

a. On the Ground Model

This model was built on the *Weight.on.Wheels* = 1 data subset.

(1) Correct classification rate. This model has a correct classification rate of 87.64% (532/607).

(2) Coincidence matrix. The matrix shows that the model can not classify regimes 3 and 4 very well. In these two regimes, the aircraft is executing a taxi turn to left/right. The same problem is also very evident for regime 5 (take-off) and regime 4.

Regimes	2	3	4	5
2	137	14	5	10
3	3	167	0	0
4	18	2	165	0
5	3	18	2	63

Table 12. The Coincidence Matrix for On the Ground Model (rows show the actual)

(3) Rulesets.*Torque* is a very strong parameter in rulesets. A quick inspection offers an idea of different regime patterns. Since the subsetting process is executed using *Weight.on.Wheels* before model fitting, there is a main "if" statement at the outer most layer of the rulesets; and that is "If *Weight.on.Wheels* = 1". Inner layers of statements are created by the sub-tree. (see Appendix C Figure 49 to see a sample part of the rulesets for this model.)

b. Take-offs and Landings Model

This model was built on the *Take.off.Flag* = 1 or *Landing.Flag* = 1 subset. There is no landing regime in the experimental flight so that regime can be not predicted in the model. The *Take.off.Flag* parameter has a value of "1" when the aircraft is in a take-off regime. This values is expected in this regime, but a very small proportion of records with this value of *Take.off.Flag* fall in the next regime flown, regime 7. Regime 7 is not a member of this family, but Regime 7 is a hover regime and may be executed right after a take-off regime, so this difficulty was accepted as the nature of the flight**,** and it is not a very bad classification.

(1) Correct classification rate. This model has a correct classification rate of 85.29% (29/35).

(2) Coincidence matrix. The matrix below shows that the model can classify take-off regimes very well.

Regimes	5	7
5	27	1
7	4	2

Table 13. The Coincidence Matrix for The Take-off and Landings Model

(3) Rulesets. There is only one statement in the ruleset, that is "if Torque <= 43.907 then 5 else 7."

c. In the Air and Slow and Control Reversals Present Model

This model was built on the *Weight.on.Wheels = 0 and KCAS<44.72 and !Control.Reversal.ID=0 (no Control.Reversal.ID present)* data subset. There must be regime 15, 16, and 17 in this subset, but 15 and 17 are not observed. These regimes were flown in the flight and expected to be in this subset. Their absence is due to the *Control.Reversal.ID*: the expected values for this parameter were not observed. This caused a problem in the subsetting process, which lead to some observations being directed into wrong regime families. In this model, only observations in regime 16 are in the subset. All predictions therefore default to regime 16.The correct classification rate is 100%.

d. In the Air and Slow; No Control Reversals Present Model

This model is built on the *Weight.on.Wheels = 0 and KCAS<44.72 and Control.Reversal.ID=0* subset. No regime 5 data should be in this subset. Since some observations with of *Weight.on.Wheels = 0* are in regime 5, some proportion of observations belonging to that regime fall in this subset. Regime 26 and 27 have some observations with *KCAS<44.72*; they are also in this subset. They are bad misclassifications.

(1) Correct classification rate. This model has a correct classification rate of 92.24% (1879/2037).

Correct	1879	92.24%
Wrong	158	7.76%
Total	2037	

Table 14. The Correct Classification Rate for The In the Air And Slow; No Control Reversals Present Model

(2) Coincidence matrix. The matrix (see Table 15) shows that the model is recognizing regimes very well. There are some misclassifications between in-ground-effect hover (regime 7) and out-of-ground-effect hover (regime 8), but they are not bad misclassifications. Regime 7 was also classified as regime 15, 16 and 17. Actually, regimes 15, 16 and 17 should

have had control reversals, but *Control.Reversals.ID* was not observed for any of those regimes. That's why the observations of those regimes were directed into this data subset. Because a regime which contains a reversal (perhaps by an evasive maneuver) can not be accepted as a normal hover regime, even if they were in the same family, these misclassifications are bad misclassifications (see Table 15.)

Regimes	10	11	12	13	14	15	16	17	26	27	28	5	7	8	9
10	**161**	0	1	2	4	0	0	0	0	0	0	0	0	0	0
11	0	**162**	0	3	0	0	0	0	0	0	0	0	0	0	2
12	2	0	**154**	4	3	0	0	0	0	0	0	0	4	1	0
13	1	2	7	**151**	2	0	0	0	0	0	0	0	0	0	4
14	0	0	0	0	**152**	0	0	0	0	0	0	0	1	0	1
15	0	0	0	0	0	**167**	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	**153**	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	**167**	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	**73**	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	**58**	0	0	0	0	0
28	0	0	0	0	0	0	0	0	1	0	**6**	0	0	0	0
5	0	0	1	0	1	0	0	0	0	0	0	**57**	3	1	2
7	0	2	5	0	0	6	10	13	0	1	0	1	**103**	21	3
8	1	0	5	1	0	0	0	0	0	0	0	0	8	**161**	3
9	0	4	7	7	0	0	0	0	0	0	0	1	1	5	**154**

Table 15. The Coincidence matrix for the model In the Air And Slow; No Control Reversals Present

(3) Rulesets. Due to the preliminary subsetting process, there is a main "if"statement at the outer most layer of the rulesets; it is "if the *Weight.on.Wheels = 0 and KCAS<44.72 and Control.Reversal.ID=0.*"Inner layers of statements are always built by this sub-models. See Appendix C Figure 50 to see a sample portion of the rulesets for this model.

e. In the Air and Fast and Control Reversals Present

This model is built on the *Weight.on.Wheels = 0 and KCAS>=44.72 and* ***!****Control.Reversal.ID=0* subset. Only the observations with *Control.Reversal.IDs* are in this model. The other portions of these regimes' data are in the other data subset without *Control.Reversal.IDs*.

(1) Correct classification rate. This model has a correct classification rate of 100% (42/42).

Correct	42	100.00%
Wrong	0	0.00%
Total	42	

Table 16. The Correct Classification Rate for In the Air and Fast; Control Reversals Present

(2) Coincidence matrix. The matrix shows that the model is very powerful in classifying the regimes in this family.(see Table 17.)

Regime	26	27	28	45	48	50	52	54
26	2	0	0	0	0	0	0	0
27	0	6	0	0	0	0	0	0
28	0	0	3	0	0	0	0	0
45	0	0	0	9	0	0	0	0
48	0	0	0	0	5	0	0	0
50	0	0	0	0	0	9	0	0
52	0	0	0	0	0	0	2	0
54	0	0	0	0	0	0	0	6

Table 17. The Coincidence Matrix for the Model In the Air and Fast; Control Reversals Present

(3) Rulesets. There is a main "if-statement" at the outer most layer of the rulesets; it is "the *Weight.on.Wheels = 0 and KCAS>=44.72 and !Control.Reversal.ID=0.*" Inner layers of statements are built on this one. See Appendix C Figure 51 to see a sample part of the rulesets for this model.

f. In the Air and Fast and No Control Reversals Present

This model is built on the *Weight.on.Wheels* = 0 and *KCAS>=44.72 and Control.Reversal.ID=0* subset. There are some observations that should have control reversals but in the data they do not. When *Control.Reversal.ID* was used as the filtering parameter, those observations were directed into this family. This is a natural result of the data, but they lead to bad classifications. Another bad classification is regime 9. There are some observations in regime 9 that have *KCAS* values greater than 44.72.

Correct	5785	99.45%
Wrong	32	0.55%
Total	5817	

Table 18. The Correct Classification Rate for In the Air And Fast; No Control Reversals Present

(2) The Coincidence matrix. The matrix shows that the model is very powerful in classifying the regimes in this family.

	19	20	21	22	23	24	25	26	27	28	36	37	40	41	42	43	44	45	46	48	49	5	50	51	52	53	54	55	56	57	59	60	61	63	64	65	9
19	149	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	12	156	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	150	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	176	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	175	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	164	0	0	2	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	168	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	112	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	121	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	12	160	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	169	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	0	169	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	176	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
41	0	0	0	0	0	0	0	0	0	0	0	0	2	176	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	162	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	173	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	168	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	177	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	168	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	168	0	0	0	0	0	0	0	0	0	0	0	0	0	0
51	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0	0	0	0	0	0	0	0	0	0	0	0
52	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0	0	0	0	0	0	0	0	0	0	0
53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0	0	0	0	0	0	0	0	0	0
54	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	171	0	0	0	0	0	0	0	0	0	0
55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	168	0	0	0	0	0	0	0	0	0
56	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	166	0	0	0	0	0	0	0	0
57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	171	0	0	0	0	0	0	0
59	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	171	0	0	0	0	0	0
60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	169	0	0	0	0	0
61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0	0	0
63	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	170	0	0	0
64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0
65	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	168	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6

Table 19. The Coincidence Matrix for the Model In the Air And Fast; No Control Reversals Present

(3) Rulesets. There is a main “if” statement at the outer most layer of the rulesets; it is “if the *Weight.on.Wheels = 0 and KCAS>=44.72 and Control.Reversal.ID=0*”. The inner layers of statements are built by this model. See Appendix C Figure 52 for a sample portion of the rulesets for this model.

2. The Analysis and Results of the Rpart Model

This model was built using the Rpart library in the S-Plus software package. There are three sub-models fitted to three different subsets, and the coincidence matrices are formed using the test set. There are two coincidence

matrices for each model; one is from the initial model fitting without loss matrix, and the second one is from the same model fitting with loss matrix. Here only the second one will be given. In fact these two matrices are not very different. The loss matrix was formed by using the predicted values of the test set. When deciding on the penalty for each misclassification, the process tries to focus only on the bad misclassifications.

a. On the Ground Model

This model is built on the *Weight.on.Wheels* = 1 data subset.

(1) Correct classification rate. This model has a correct classification rate of 89.8% (545/607) (see Table 20.)

Regimes	2	3	4	5
2	141	11	4	10
3	8	158	3	1
4	0	3	182	0
5	8	1	13	64

Table 20. The Coincidence Matrix of the On the Ground Model in Rpart

(2) Coincidence matrix. The matrix shows that the model is having the same problems as the C5.0.The regime 2 and 3 are misclassified fairly often. The same problem also exists for regime 4 and 5 (see Table20).

(3) Classification tree. The simplified version of the tree is given in Appendix C Figure 53.

b. In the Air and Slow Model

This model is built on the Weight.on.Wheels = 0 and KCAS<44.72 data subset.

(1) Correct classification rate. This model has a correct classification rate of 92.6% (1916/2069) (see Table 21.)

	10	11	12	13	14	15	16	17	26	27	28	5	7	8	9	Total
10	151	3	4	4	2	0	0	0	0	0	0	0	0	3	1	168
11	1	161	0	4	0	0	0	0	0	0	0	0	0	0	1	167
12	0	3	127	9	0	0	0	0	0	0	0	0	6	4	19	168
13	3	3	0	161	0	0	0	0	0	0	0	0	0	0	0	167
14	2	0	3	0	149	0	0	0	0	0	0	0	0	0	0	154
15	0	0	0	0	0	157	10	0	0	0	0	0	0	0	0	167
16	0	2	0	0	0	0	160	5	0	0	0	0	0	0	0	167
17	0	0	0	0	0	4	2	161	0	0	0	0	0	0	0	167
26	0	0	0	0	0	0	0	0	84	0	0	0	0	0	0	84
27	0	0	0	0	0	0	0	0	0	58	0	0	0	0	0	58
28	0	0	0	0	0	0	0	0	2	0	5	0	0	0	0	7
5	2	1	1	1	1	0	0	0	0	0	0	60	0	0	0	66
7	0	1	3	0	0	0	0	0	0	0	0	7	142	12	0	165
8	0	0	5	0	0	1	0	0	0	0	0	0	8	164	1	179
9	0	0	1	4	1	0	0	0	0	0	0	0	0	3	176	185
																2069
diag	151	161	127	161	149	157	160	161	84	58	5	60	142	164	176	1916

Correct Classification Rate 0.926

Table 21. The Summary of the In the Air And Slow Model in Rpart

(2) Coincidence matrix. The matrix shows that the model has problems in classifying the hover regimes. The misclassifications are not big in numbers nor bad classifications (see Table 21.)

(3) Classification tree. The simplified version of the tree is given in Appendix C Figure 54.

b. In the Air and Fast Model

This model is built on the Weight.on.Wheels = 0 and KCAS>=44.72 data subset.

(1) Correct classification rate. This model has a correct classification rate of 97.5% (5654/5799) (see Table 22.)

	19	20	21	22	23	24	25	55	59	60	26	28	27	56	36	37	40	41	42	43	44	45	46	48	49	5	50	51	52	53	54	57	61	63	64	65	Total
19	150	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	150
20	23	145	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	168
21	0	0	150	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	150
22	0	0	0	169	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	176
23	0	0	0	1	170	0	0	0	0	0	0	0	0	0	0	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	175
24	0	0	0	0	0	150	0	0	1	17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	168
25	0	0	0	0	0	0	168	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	168
55	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	101
59	0	0	0	0	0	1	0	0	117	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	121
60	0	9	0	0	0	0	0	4	8	151	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	172
26	0	0	0	0	0	0	0	0	0	0	169	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	169
28	0	0	0	0	0	0	0	0	0	0	0	166	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0	0	0	0	169
27	0	0	0	0	0	0	0	0	0	0	0	0	176	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	176
56	0	0	0	0	0	0	0	0	0	0	0	0	0	178	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	178
36	0	0	0	5	0	0	0	0	0	0	0	0	0	0	157	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	162
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	171	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	173
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167
42	0	2	0	1	0	0	0	0	0	0	0	0	0	0	0	6	0	3	156	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	168
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	177	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	177
44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	156	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	168
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	18
46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	0	156	0	0	0	0	0	0	0	0	0	0	0	0	0	168
48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0	0	0	0	0	0	0	0	0	0	0	167
49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	166	0	1	0	0	0	0	0	0	0	0	0	167
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0	0	0	0	0	0	0	0	0	167
50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	171	0	0	0	0	0	0	0	0	0	171
51	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	161	0	0	0	0	5	0	0	0	168
52	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	162	0	0	0	2	0	0	0	167
53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0	0	0	0	4	171
54	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	171	0	0	0	0	0	171
57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	167	0	0	0	0	169
61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	0	0	167
63	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	170	0	0	170
64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	167	0	167
65	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	167	168
																																					5799
diag	150	145	150	169	170	150	168	100	117	151	169	166	176	178	157	171	167	167	156	177	156	18	156	167	166	167	171	161	162	167	171	167	167	170	167	167	5654

Correct Classification Rate **0.974996**

Table 22. The Summary of the In the Air And Fast Model in Rpart

(2) Coincidence matrix. The matrix shows that the model has great power in classifying these regimes. The parameter values are more distinct for the various regimes, and this aids the algorithm in classification (see Table 22.)

(3) Classification tree. The simplified version of the tree is given in Appendix C Figure 55.

3. Finding the Correct Classification Rate for Future Predictions

To find the correct classification rates for prediction using new data sets, a weighted average of the rates should be calculated for both sets of models. The distribution of the regime families that might be seen in a randomly selected-flight are used as the weights for averaging the correct classification rates. The number of each regime in the predicted regime vector (produced by a model) is counted to extract a probability distribution for each regime family. An example is given in the following table.

The C5.0 MODELS	Correct Classification Rates	A Possible Distribution of Predicted Regime Families
The On the Ground Sub-Model	87.6%	0.1
The Take-offs and Landings Sub-Model	85.3%	0.05
The In the Air and Slow with CR Sub-Model	100.0%	0.05
The In the Air and Slow w/o CR Sub-Model	92.2%	0.1
The In the Air and Fast with CR Sub-Model	100.0%	0.05
The In the Air and Fast w/o CR Sub-Model	99.5%	0.65
The Overall Average Correct Classification	96.9%	

The RPART MODELS	Correct Classification Rates	A Possible Distribution of Predicted Regime Families
The On the Ground Sub-Model	89.0%	0.1
The In the Air and Slow Sub-Model	92.6%	0.2
The In the Air and Fast Sub-Model	97.5%	0.7
The Overall Average Correct Classification	95.7%	

Table 23. Finding the Correct Classification Rate for Predictions

For the example given in the table above, the overall correct classification achieved by C5.0 is 96.9%, and for Rpart model the correct classification rate is 95.7%.

4. The Overall Correct Classification Rate Achieved By This Study

The achieved overall correct classification is over 95%. The distribution for the regime families are extracted from the training dataset. Naturally, the distribution of regime families of a randomly-selected flight may be very different than the one in the training set. Here, the number of observations for each regime flown was the same in the training dataset. The calculation is given in the table below.

The C5.0 MODELS	Correct Classification Rates	The Distribution of Regime Families in The Training Data
The On the Ground Sub-Model	87.6%	0.1
The Take-offs and Landings Sub-Model	85.3%	0.02
The In the Air and Slow with CR Sub-Model	100.0%	0.06
The In the Air and Slow w/o CR Sub-Model	92.2%	0.16
The In the Air and Fast with CR Sub-Model	100.0%	0.26
The In the Air and Fast w/o CR Sub-Model	99.5%	0.4
The Overall Average Correct Classification	97.0%	

The RPART MODELS	Correct Classification Rates	The Distribution of Regime Families in The Training Data
The On the Ground Sub-Model	89.0%	0.1
The In the Air and Slow Sub-Model	92.6%	0.24
The In the Air and Fast Sub-Model	97.5%	0.66
The Overall Average Correct Classification	95.5%	

Table 24. The Overall Correct Classification Rate Achieved by This Study

B. CONCLUSION

The purpose of this study was to build a model that predicts the flight regimes. Models were chosen to produce as few as bad misclassifications. When a single model was built on the original data, there were numerous bad misclassifications and some important parameters were not used at the appropriate branches as the parameters on which to split. To prevent these problems, the data was divided into smaller sets using important and very distinctive parameters to make sure that they contribute to the model at the correct step. Out of many options, C5.0 and Rpart algorithms produced the superior results. By giving more attention to these two models, better results were achieved. The approach was the same for both models: Filtering or rounding some parameters to mute the uninteresting values, fitting sub-trees to subsets, pruning the trees as much as possible, using a test set to form predicted values to obtain the correct classification rates, and inspecting the rulesets or

classification trees to determine whether or not they produced valid physical rules. Both models have nearly the same problems in classifying low-speed regimes, and they have great power in classifying high-speed regimes. The overall performance of the models was nearly the same. When interpreting the model, using the ruleset may be easier than using the classification trees. For both models, preliminary partitioning using the important parameters ensures the minimum number of bad misclassifications. This approach also guarantees that the most of the misclassifications are within a regime family.

A future study may focus on the sensitivity analysis of the classification models. A possible research question might be “how good is the model at predicting regimes of an independent flight?” The flight data might be preferably collected in various conditions, such as with varying weight of the aircraft and under significantly different weather conditions.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX A. THE COINCIDENCE MATRICES for C&RT AND CHAID

	10	12	13	15	2	20	25	3	36	37	4	54	55	56	57	59	63	7	8
10	230	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	212	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	132	0
12	0	209	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	59
13	0	0	300	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9
14	2	174	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	88	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	71	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	311	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	805	0	0	0	0	0	21	0	0	0	0	0	0	0	0
20	0	0	0	0	0	572	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	323	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	293	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	287	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	598	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	590	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	199	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	211	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	233	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	215	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	596	0	0	0	0	0	0	0	0	0	0
37	0	0	0	0	0	0	0	0	0	621	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	196	0	0	0	0	0	0	0	0
40	0	0	0	0	0	318	0	0	0	0	0	0	0	0	0	0	0	0	0
41	0	0	0	0	0	299	0	0	0	0	0	0	0	0	0	0	0	0	0
42	0	0	0	0	0	108	0	0	0	0	0	0	0	0	0	0	0	0	0
43	0	0	0	0	0	371	0	0	0	0	0	0	0	0	0	0	0	0	0
44	0	0	0	0	0	144	0	0	0	0	0	0	0	0	0	0	0	0	0
45	0	0	0	0	0	99	0	0	0	0	0	0	0	0	0	0	0	0	0
46	0	0	0	0	0	288	0	0	0	0	0	0	0	0	0	19	0	0	0
48	0	0	0	0	0	143	0	0	0	0	0	0	0	0	0	0	0	0	0
49	0	0	0	0	0	160	0	0	0	0	0	0	0	0	0	0	0	0	0
5	2	76	0	0	0	0	0	197	0	0	118	0	0	0	0	0	0	79	1
50	0	0	0	0	0	166	0	0	0	0	0	0	0	0	0	0	0	0	0
51	0	0	0	0	0	0	0	0	0	0	0	75	0	0	0	0	0	0	0
52	0	0	0	0	0	0	0	0	0	0	0	88	0	0	0	0	0	0	0
53	0	0	0	0	0	0	0	0	0	0	0	84	0	0	0	0	0	0	0
54	0	0	0	0	0	0	0	0	0	0	0	190	0	0	0	0	0	0	0
55	0	0	0	0	0	0	0	0	0	0	0	0	843	0	0	0	0	0	0
56	0	0	0	0	0	0	0	0	0	0	0	0	20	617	0	0	0	0	0
57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	456	0	0	0	0
59	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	444	0	0	0
60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	363	0	0	0
61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	188	0	0	0
63	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	433	0	0
64	0	0	0	0	0	0	0	0	0	0	0	0	0	95	0	0	0	0	0
65	0	0	0	0	0	0	0	0	0	0	0	0	0	0	238	0	0	0	0
7	7	72	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1,185	108
8	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	387
9	0	194	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	22

Figure 45. The Coincidence Matrix of the C&RT Model

	10	11	12	13	14	15	19	2	20	21	22	23	24	25	26	3	36	37	4	40	41	43	48	5	50	54	55	56	57	59	60	61	63	65	7	8
10		64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	166		246	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	62	0		191	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	36	0
13	0	0	0		161	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	59	0
14	0	0	0	0		149	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	148
15	0	0	25	0	0		38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0
16	0	19	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0
17	0	31	0	0	0	0	28		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	6	0
19	0	27	0	0	0	0	4	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	2	0	133	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0	0
43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0	0
44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0	0
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0	0
46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0	0
48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0	0
49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0	0
51	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0	0
52	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0	0
53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0	0
54	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0	0
55	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		0
56	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
57	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
59	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
60	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
63	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
65	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	83	2	0	0	19	6	9	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1,242	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	403
9	0	0	144	0	48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24	0

Figure 46. The Coincidence Matrix of the CHAID Model

APPENDIX B. THE CLEMENTINE TRAINING AND TEST STREAMS

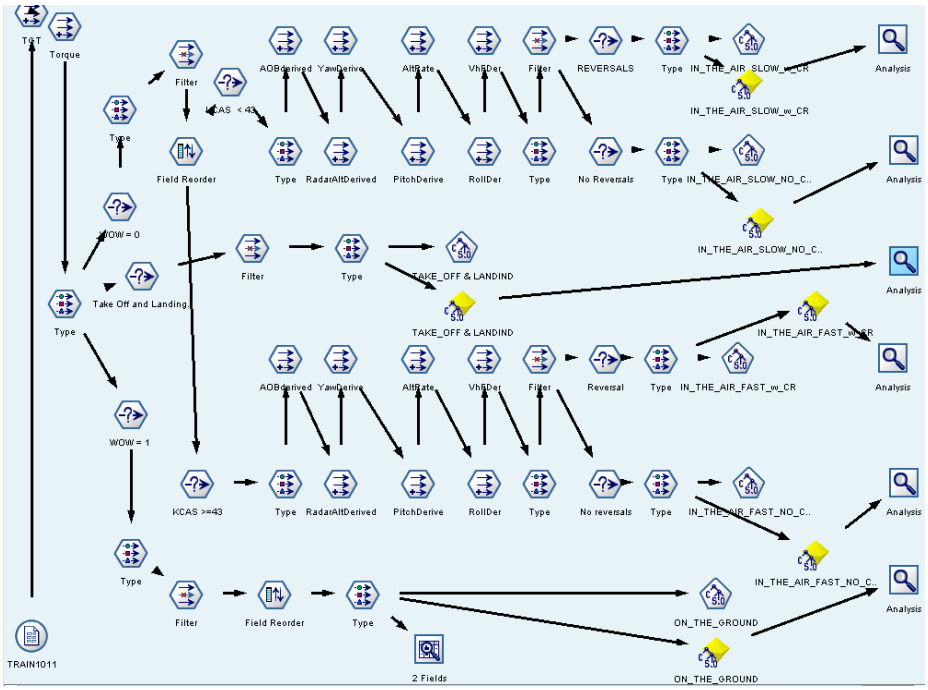


Figure 47. The Clementine Training Stream

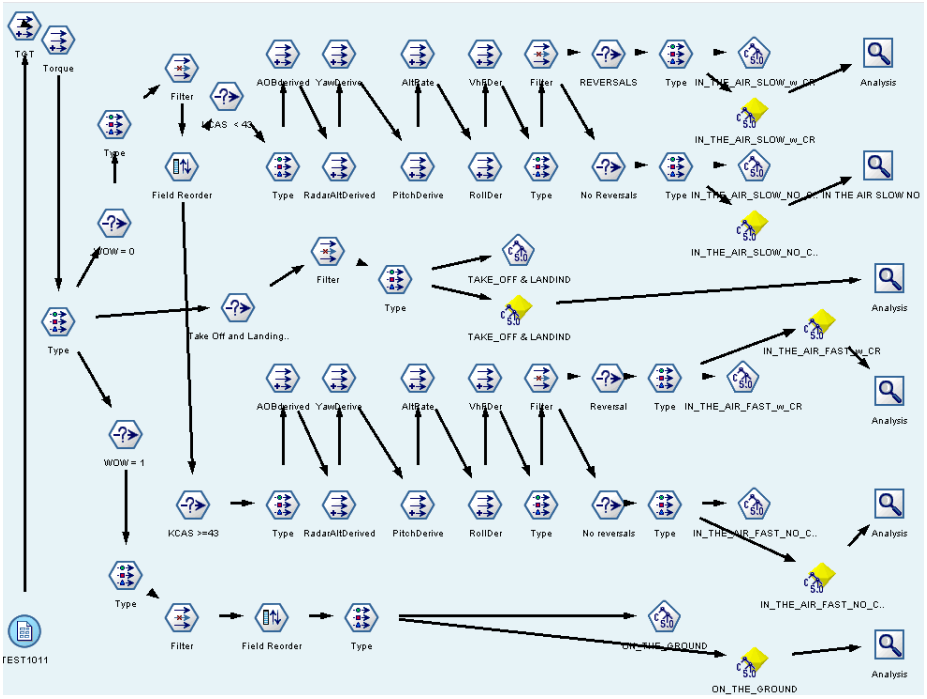


Figure 48. The Clementine Test Stream

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C. SAMPLES FROM RULESETS FROM C5.0

```
Ruleset 1: ON THE GROUND
    Rules for 2 - contains 2 rule(s)
        Rule 1 for 2
            if Yawrate > -0.647 and Torque > 22.814 and Torque <= 24.190 and Nr <= 101.262 then 2
        Rule 2 for 2
            if Yawrate > -0.771 and Torque <= 24.620 then 2
    Rules for 3 - contains 8 rule(s)
        Rule 1 for 3
            if Torque > 19.878 and Torque <= 22.329 and Nr > 100.028 and Nr <= 100.537 then 3
        Rule 2 for 3
            if Nr <= 99.458 then 3
        Rule 3 for 3
            if Yawrate <= -0.220 and Torque > 18.528 and Torque <= 19.484 and Nr > 100.654 then 3
        Rule 4 for 3
            if Yawrate <= -0.647 and Torque > 22.814 and Torque <= 24.190 and Nr <= 101.522 then 3
        Rule 5 for 3
            if Yawrate <= 0.642 and Torque > 17.947 and Nr <= 100.028 then 3
        Rule 6 for 3
            if Yawrate > -0.468 and Yawrate <= 0.642 and Torque > 19.484 and Torque <= 21.162 and Nr > 100.028 and Nr <= 100.700 then 3
        Rule 7 for 3
            if Torque > 17.947 and Torque <= 18.364 and Nr > 100.607 then 3
        Rule 8 for 3
            if Torque > 17.660 then 3
    Rules for 4 - contains 9 rule(s)
        Rule 1 for 4
            if Yawrate > 0.642 then 4
        Rule 2 for 4
            if Yawrate > -0.564 and Torque > 18.184 and Torque <= 18.448 and Nr <= 100.607 then 4
        Rule 3 for 4
            if Yawrate > -0.564 and Yawrate <= -0.457 and Torque > 17.969 and Torque <= 18.122 and Nr <= 100.607 then 4
        Rule 4 for 4
            if Yawrate <= -0.564 and Torque > 17.969 and Torque <= 18.528 and Nr <= 100.376 then 4
        Rule 5 for 4
            if Yawrate > -0.371 and Torque > 17.969 and Torque <= 18.122 and Nr > 100.028 then 4
        Rule 6 for 4
            if Torque > 17.969 and Torque <= 18.122 and Nr > 100.028 and Nr <= 100.376 then 4
        Rule 7 for 4
            if Torque > 18.528 and Torque <= 19.484 and Nr > 100.607 and Nr <= 100.654 then 4
        Rule 8 for 4
            if Yawrate > -0.220 and Torque <= 19.484 and Nr > 100.537 then 4
        Rule 9 for 4
            if Yawrate > -3.140 and Torque > 21.162 and Torque <= 24.190 then 4
    Rules for 5 - contains 16 rule(s)
        Rule 1 for 5
            if Torque > 24.190 and Nr > 101.592 then 5
        Rule 2 for 5
            if Yawrate > -0.771 and Torque > 24.620 then 5
        Rule 3 for 5
            if Yawrate <= -0.468 and Torque > 19.484 and Torque <= 21.162 and Nr > 100.537 then 5
        Rule 4 for 5
            if Yawrate <= 0.642 and Torque > 19.484 and Torque <= 21.162 and Nr > 100.700 then 5
        Rule 5 for 5
            if Torque > 17.632 and Torque <= 17.660 and Nr > 100.122 then 5
        Rule 6 for 5
            if Yawrate <= -0.620 and Torque <= 22.814 and Nr > 101.238 then 5
        Rule 7 for 5
            if Yawrate <= -2.347 and Torque > 24.190 then 5
        Rule 8 for 5
            if Yawrate <= -0.564 and Torque <= 18.528 and Nr > 100.376 then 5
        Rule 9 for 5
            if Torque > 18.448 and Torque <= 18.528 and Nr > 100.028 and Nr <= 100.607 then 5
        Rule 10 for 5
            if Yawrate <= -0.647 and Nr > 101.522 then 5
        Rule 11 for 5
            if Yawrate <= -0.444 and Torque > 17.632 and Torque <= 17.727 and Nr > 100.122 then 5
        Rule 12 for 5
            if Torque > 17.632 and Torque <= 17.727 and Nr > 100.420 then 5
        Rule 13 for 5
            if Torque > 17.574 and Torque <= 17.585 and Nr > 100.122 then 5
        Rule 14 for 5
            if Yawrate <= -0.400 and Torque > 17.823 and Torque <= 17.947 and Nr > 100.122 and Nr <= 100.329 then 5
        Rule 15 for 5
            if Torque > 17.947 and Torque <= 17.969 then 5
        Rule 16 for 5
            if Torque > 18.364 and Torque <= 18.528 and Nr > 100.607 then 5
    Default: 2
```

Figure 49. A Sample Part of the Ruleset for the On the Ground Model

```
Ruleset 1:IN THE AIR SLOW WITH NO CR
  Rules for 5 - contains 13 rule(s)
      Rule 1 for 5
          if Torque > 12.302 and Torque <= 39.023 then 5
      Rule 2 for 5
          if Lateral.Accel > 0.096 and Torque <= 43.444 and Nr > 100.443 and AltRate > -566.809 then 5
      Rule 3 for 5
          if Torque > 43.444 and Torque <= 47.408 and RollDer > -4.010 then 5
      Rule 4 for 5
          if Vertical.Accel > 1.020 and Torque > 62.433 and AltRate > 0 then 5
      Rule 5 for 5
          if Vertical.Accel > 1.090 and Torque > 56.565 and Nr > 100.283 and RollDer > -0.270 then 5
      Rule 6 for 5
          if Torque <= 54.951 and AltRate > 444.192 and AltRate <= 476.131 then 5
      Rule 7 for 5
          if Lateral.Accel > 0.101 and Torque > 47.408 and Torque <= 51.919 and Nr <= 100.399 and AltRate <= 589.939 and RollDer > -4.456 then 5
      Rule 8 for 5
          if Lateral.Accel > 0.098 and Vertical.Accel > 1.057 and Torque > 56.565 and Nr > 100.376 and RollDer > -4.010 then 5
      Rule 9 for 5
          if Torque > 53.412 and Torque <= 53.543 and Nr > 100.467 and AltRate > -493.936 then 5
      Rule 10 for 5
          if Torque > 57.541 and Torque <= 58.031 and Nr > 100.122 and Nr <= 100.189 and AltRate <= 0 and RollDer > -0.142 then 5
      Rule 11 for 5
          if Lateral.Accel <= 0.098 and Vertical.Accel > 0.964 and Vertical.Accel <= 0.997 and Torque > 58.842 and Torque <= 59.589 and Nr > 100.607 then 5
      Rule 12 for 5
          if Lateral.Accel > 0.098 and Torque > 59.435 and Torque <= 59.974 and Nr > 100.654 and Nr <= 100.911 and RollDer > -4.010 then 5
      Rule 13 for 5
          if Lateral.Accel > 0.088 and Vertical.Accel > 1.025 and Torque <= 53.116 and AltRate > -493.936 and AltRate <= 444.192 then 5
  Rules for 7 - contains 24 rule(s)
      Rule 1 for 7
          if Torque <= 56.565 and AltRate > -589.300 and AltRate <= -493.936 then 7
      Rule 2 for 7
          if Torque > 51.919 and Torque <= 53.819 and Nr > 100.307 and Nr <= 100.467 and RollDer > -4.010 then 7
      Rule 3 for 7
          if Lateral.Accel <= 0.117 and Torque > 54.951 and Torque <= 56.565 and Nr > 99.892 and Nr <= 100.028 then 7
      Rule 4 for 7
          if Torque > 55.697 and Torque <= 56.380 and Nr > 100.259 and Nr <= 100.353 and AltRate > -593.574 and RollDer > -5.055 then 7
      Rule 5 for 7
          if Lateral.Accel <= 0.080 and Torque > 54.014 and Torque <= 54.925 and AltRate <= 412.704 then 7
      Rule 6 for 7
          if Torque > 54.951 and Torque <= 56.565 and Nr > 100.352 and AltRate > 434.448 then 7
      Rule 7 for 7
          if Torque > 54.951 and Torque <= 55.316 and RollDer > -4.559 and RollDer <= -0.028 then 7
      Rule 8 for 7
          if Vertical.Accel <= 1.034 and Torque > 51.919 and Torque <= 55.697 and Nr > 100.398 and Nr <= 100.422 and AltRate > -404.419 and RollDer > -5.055 then 7
      Rule 9 for 7
          if Torque > 51.919 and Torque <= 56.565 and RollDer > 0.033 then 7
      Rule 10 for 7
          if Torque > 54.951 and Torque <= 55.697 and Nr > 100.724 then 7
      Rule 11 for 7
          if Vertical.Accel > 0.970 and Vertical.Accel <= 0.974 and Torque > 55.697 and Torque <= 56.565 and Nr <= 100.770 and AltRate <= 0 and RollDer > -5.055 then 7
      Rule 12 for 7
          if Torque > 56.031 and Torque <= 56.565 and Nr > 100.537 and AltRate > -493.936 and RollDer <= -0.270 then 7
      Rule 13 for 7
          if Torque > 52.995 and Torque <= 53.166 and Nr <= 100.306 then 7
      Rule 14 for 7
          if Vertical.Accel > 0.974 and Vertical.Accel <= 0.983 and Torque > 53.819 and Torque <= 54.951 and Nr > 100.306 and Nr <= 100.376 and RollDer > -4.632 then 7
      Rule 15 for 7
          if Lateral.Accel <= 0.080 and Vertical.Accel > 1.012 and Torque > 54.951 and Torque <= 55.697 then 7
      Rule 16 for 7
          if Vertical.Accel <= 0.974 and Torque > 55.381 and Torque <= 55.667 and Nr > 100.491 and RollDer > -0.028 then 7
      Rule 17 for 7
          if Torque > 54.951 and Torque <= 55.116 and Nr > 100.422 and RollDer > -5.055 then 7
      Rule 18 for 7
          if Torque > 55.459 and Torque <= 55.549 and Nr > 100.422 and RollDer > -0.028 then 7
      Rule 19 for 7
          if Lateral.Accel > 0.083 and Torque > 55.372 and Torque <= 56.565 and Nr > 100.075 and Nr <= 100.237 and AltRate <= 589.939 and RollDer > -0.352 then 7
      Rule 20 for 7
          if Lateral.Accel > 0.080 and Lateral.Accel <= 0.147 and Vertical.Accel > 0.974 and Torque > 54.951 and Torque <= 55.697 and Nr > 100.491 and Nr <= 100.537 then 7
      Rule 21 for 7
          if Lateral.Accel > 0.126 and Vertical.Accel > 0.997 and Torque > 57.019 and Torque <= 58.031 and Nr <= 100.189 and RollDer > -0.142 then 7
```

Figure 50. A Sample Part of the Ruleset for the In the Air and Slow With No Control Reversals Model

```
Ruleset 1:IN THE AIR AND FAST WITH CR
  Rules for 26 - contains 1 rule(s)
      Rule 1 for 26
          if CONTROL.REVERSAL.ID = 2 and AltRate > -1765.250 then 26
  Rules for 27 - contains 1 rule(s)
      Rule 1 for 27
          if CONTROL.REVERSAL.ID = 4 and AltRate > -1765.250 then 27
  Rules for 28 - contains 1 rule(s)
      Rule 1 for 28
          if CONTROL.REVERSAL.ID = 8 then 28
  Rules for 45 - contains 1 rule(s)
      Rule 1 for 45
          if AltRate <= -1765.250 and VhFDer <= 0.496 then 45
  Rules for 48 - contains 1 rule(s)
      Rule 1 for 48
          if CONTROL.REVERSAL.ID = 2 and PitchDerive > -6.641 and AltRate <= -1765.250 then 48
  Rules for 50 - contains 1 rule(s)
      Rule 1 for 50
          if CONTROL.REVERSAL.ID = 4 and PitchDerive > -6.641 and AltRate <= -1765.250 and VhFDer > 0.496 then 50
  Rules for 52 - contains 1 rule(s)
      Rule 1 for 52
          if CONTROL.REVERSAL.ID = 2 and PitchDerive <= -6.641 then 52
  Rules for 54 - contains 1 rule(s)
      Rule 1 for 54
          if CONTROL.REVERSAL.ID = 4 and PitchDerive <= -6.641 then 54
  Default: 45
```

Figure 51. A Sample Part of the Ruleset for the In the Air and Fast with Control Reversals Model

```
Ruleset 1: IN THE AIR AND FAST WITH NO CR
  Rules for 5 - contains 1 rule(s)
      Rule 1 for 5
          if AOBderived <= 8.559 and PitchDerive <= -3.905 and Torque > 60.618 and VhFDer <= 0.805 then 5
  Rules for 9 - contains 2 rule(s)
      Rule 1 for 9
          if RollDer > -7.431 and Torque > 53.734 and Torque <= 60.618 and VhFDer <= 0.485 then 9
      Rule 2 for 9
          if PitchDerive > -3.905 and RollDer > -7.431 and Torque > 60.618 and VhFDer <= 0.394 then 9
  Rules for 19 - contains 5 rule(s)
      Rule 1 for 19
          if AltRate > -613.367 and RollDer > -7.431 and Torque <= 32.670 and VhFDer <= 0.383 then 19
      Rule 2 for 19
          if AltRate > -613.367 and RollDer > -7.431 and RollDer <= 10.585 and Torque <= 39.084 and VhFDer <= 0.365 then 19
      Rule 3 for 19
          if RollDer > -7.431 and Torque <= 33.675 and VhFDer > 0.379 and VhFDer <= 0.383 then 19
      Rule 4 for 19
          if RollDer > -7.431 and Torque <= 33.347 and Vertical.Accel > 0.964 and VhFDer > 0.370 and VhFDer <= 0.383 then 19
      Rule 5 for 19
          if RollDer <= 10.585 and Torque > 33.109 and Torque <= 39.084 and VhFDer <= 0.370 then 19
  Rules for 20 - contains 6 rule(s)
      Rule 1 for 20
          if RollDer <= 10.585 and Torque > 36.772 and Torque <= 39.084 and VhFDer <= 0.516 then 20
      Rule 2 for 20
          if RollDer > -7.431 and RollDer <= 10.585 and VhFDer > 0.383 and VhFDer <= 0.416 then 20
      Rule 3 for 20
          if Torque > 32.670 and Torque <= 33.109 and VhFDer > 0.365 and VhFDer <= 0.370 then 20
      Rule 4 for 20
          if RollDer > -7.431 and RollDer <= 10.585 and Torque > 33.675 and Torque <= 39.084 and VhFDer <= 0.383 then 20
      Rule 5 for 20
          if RollDer > -7.431 and RollDer <= 10.585 and Torque > 33.347 and VhFDer > 0.370 and VhFDer <= 0.379 then 20
      Rule 6 for 20
          if Torque > 32.670 and Vertical.Accel <= 0.964 and VhFDer > 0.365 and VhFDer <= 0.379 then 20
  Rules for 21 - contains 1 rule(s)
      Rule 1 for 21
          if AOBderived <= 8.559 and Torque > 39.084 and Torque <= 43.731 and VhFDer <= 0.594 then 21
  Rules for 22 - contains 1 rule(s)
      Rule 1 for 22
          if AOBderived <= 7.367 and AltRate > -657.675 and PitchDerive > -3.016 and
                  RollDer <= 5.093 and Torque > 30.241 and Torque <= 39.084 and VhFDer > 0.516 and VhFDer <= 0.632 then 22
  Rules for 23 - contains 2 rule(s)
      Rule 1 for 23
          if AOBderived <= 8.559 and Torque > 39.084 and Torque <= 43.731 and VhFDer > 0.594 and VhFDer <= 0.805 then 23
      Rule 2 for 23
          if AOBderived <= 13.860 and RollDer > 0 and Torque > 12.626 and Torque <= 39.084 and VhFDer > 0.632 then 23
  Rules for 24 - contains 3 rule(s)
      Rule 1 for 24
          if AOBderived <= 8.559 and Torque > 43.731 and Torque <= 60.618 and VhFDer > 0.485 and VhFDer <= 0.790 then 24
      Rule 2 for 24
          if PitchDerive <= -4.032 and Torque <= 56.954 and VhFDer > 0.790 and VhFDer <= 0.805 then 24
      Rule 3 for 24
          if AltRate > -500.654 and PitchDerive > -4.032 and PitchDerive <= -3.208 and RollDer > -7.431 and VhFDer > 0.794 and VhFDer <= 0.798 then 24
  Rules for 25 - contains 1 rule(s)
      Rule 1 for 25
          if AOBderived <= 8.559 and Torque > 52.900 and VhFDer > 0.805 and VhFDer <= 0.972 then 25
  Rules for 26 - contains 3 rule(s)
      Rule 1 for 26
          if AOBderived <= 8.559 and Torque > 60.618 and VhFDer > 0.394 and VhFDer <= 0.805 then 26
      Rule 2 for 26
          if AOBderived <= 8.559 and Torque > 48.519 and Torque <= 53.734 and VhFDer <= 0.485 then 26
      Rule 3 for 26
          if AltRate <= -613.367 and PitchDerive > 0 and RollDer > -7.431 and Torque <= 39.084 then 26
  Rules for 27 - contains 5 rule(s)
      Rule 1 for 27
          if AOBderived <= 5.092 and AltRate > -613.367 and Torque <= 36.772 and VhFDer > 0.445 and VhFDer <= 0.516 then 27
      Rule 2 for 27
          if PitchDerive > -3.208 and Torque <= 60.618 and VhFDer > 0.790 and VhFDer <= 0.805 then 27
      Rule 3 for 27
          if AOBderived <= 8.559 and PitchDerive > -4.032 and VhFDer > 0.802 and VhFDer <= 0.805 then 27
      Rule 4 for 27
          if AOBderived <= 8.559 and PitchDerive > -4.032 and RollDer > 1.159 and VhFDer > 0.790 then 27
```

Figure 52. A Sample Part of the Ruleset for the In the Air and Fast No Control Reversals Model

APPENDIX D. PLOTS OF THE CLASSIFICATION TREES BUILT USING RPART.

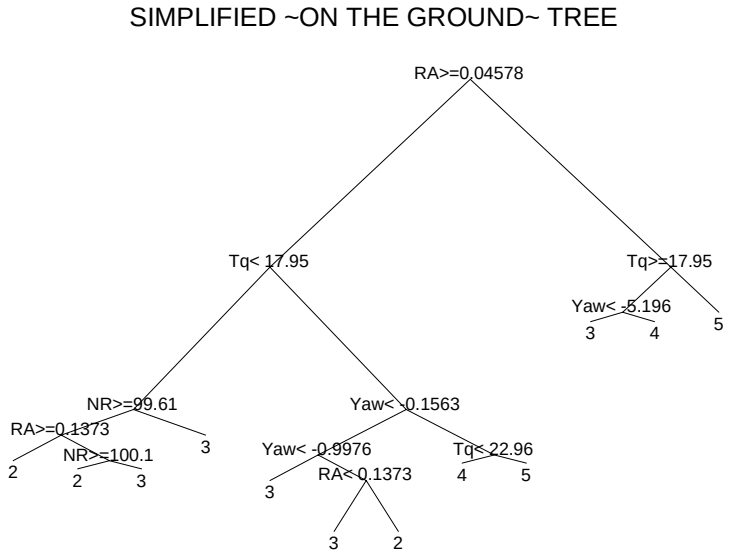


Figure 53. On the Ground Tree

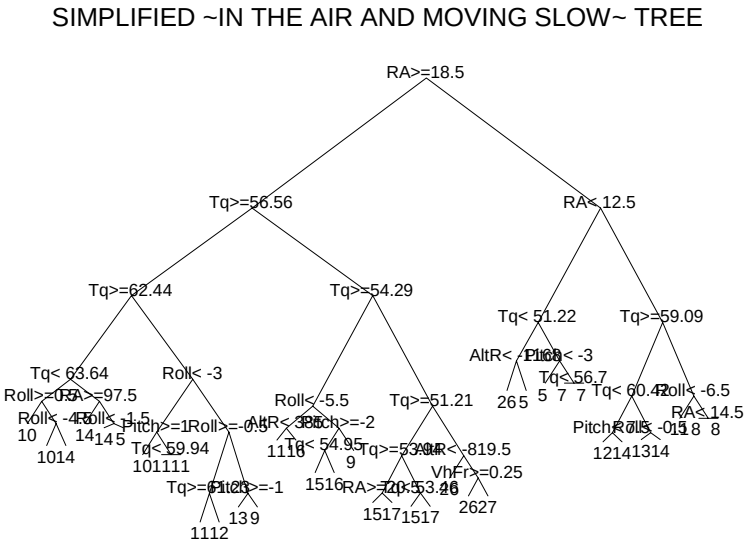


Figure 54. In the Air and Moving Slow Tree

SIMPLIFIED ~IN THE AIR AND MOVING FAST~ TREE

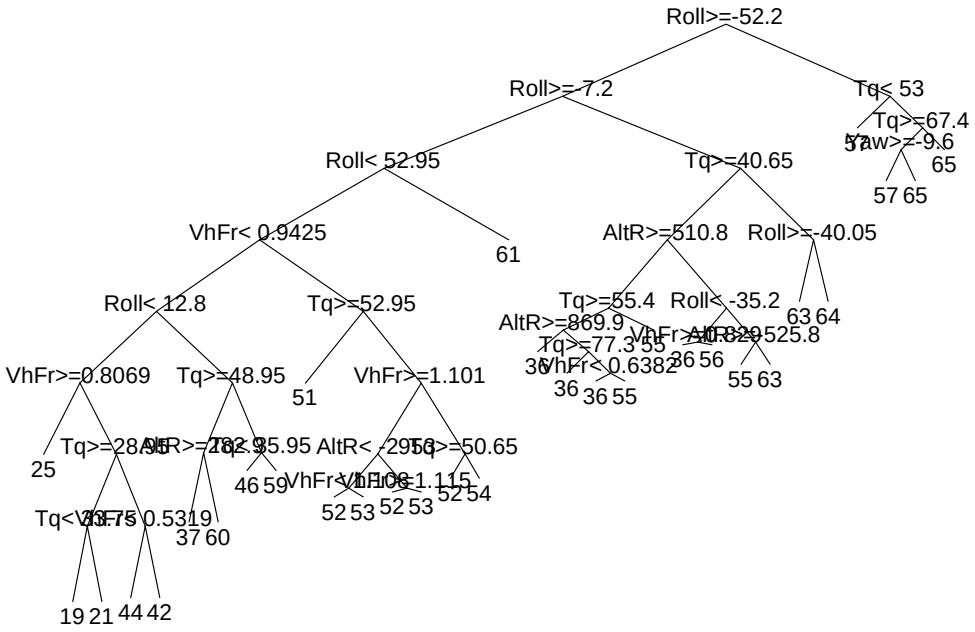


Figure 55. In the Air and Moving Fast Tree

APPENDIX E. THE SCRIPT FOR BUILDING TREE MODELS IN S-PLUS

#(This portion is only for building In the Air and slow model; the other tree codes are very similar. The differences are given at the end of this Appendix)

#####TREE BUILDING SCRIPT FOR CLASSIFICATION OF "IN THE AIR AND SLOW SPEED" REGIMES###########

```
#####################################################################################################
# Traing and testing set                                                                            .#
#####################################################################################################
TRAIN1011 <- convert.col.type(target = TRAIN1011, column.spec = list("CONTROL.REVERSAL.ID", "Takeoff.Flag", "Weight.on.Wheels", "Regime"), column.type = "factor")
menuSubset(data =TRAIN1011, subset.expression = "Weight.on.Wheels == 0", subset.columns = "<ALL>", result.type = "Data Set", subset.col.name = "Subset", save.name =
"IN.THE.AIR", show.p = SHOW.ON.SCREEN);
TEST1011 <- convert.col.type(target = TEST1011, column.spec = list("CONTROL.REVERSAL.ID", "Takeoff.Flag", "Weight.on.Wheels", "Regime"), column.type = "factor")
menuSubset(data =TEST1011, subset.expression = "Weight.on.Wheels == 0", subset.columns = "<ALL>", result.type = "Data Set", subset.col.name = "Subset", save.name =
"IN.THE.AIRT", show.p = SHOW.ON.SCREEN);

#####################################################################################################
# Transforming and rounding the parameters.Good for shortnames and unwanted digits                  .#
#####################################################################################################
TEMPTRAIN_IN.THE.AIR
TEMPTEST_IN.THE.AIRT

TEMPTRAIN <- menuTransform(data = TEMPTRAIN, variable.name = "VhFr", expression = "round(Airspeed.Vh.Fraction,1)")
TEMPTRAIN <- menuTransform(data = TEMPTRAIN, variable.name = "Tq", expression = "(Torque.1+Torque.2)/2")
TEMPTRAIN <- menuTransform(data = TEMPTRAIN, variable.name = "Roll", expression = "ifelse(Roll.Attitude>=-4&Roll.Attitude <=-2,0,round(Roll.Attitude,0))")
TEMPTRAIN <- menuTransform(data = TEMPTRAIN, variable.name = "AltR", expression = "ifelse(Altitude.Rate>=-300&Altitude.Rate <=300,0,round(Altitude.Rate,0))")
TEMPTRAIN <- menuTransform(data = TEMPTRAIN, variable.name = "Pitch", expression = "ifelse(Pitch.Attitude>=2&Pitch.Attitude <=5,0,round(Pitch.Attitude,0))")
TEMPTRAIN <- menuTransform(data = TEMPTRAIN, variable.name = "Yaw", expression = "ifelse(Yawrate>=-2&Yawrate<=2,0,round(Yawrate))")
TEMPTRAIN <- menuTransform(data = TEMPTRAIN, variable.name = "Vert", expression = "round(Vertical.Accel,1)")
TEMPTRAIN <- menuTransform(data = TEMPTRAIN, variable.name = "Lat", expression = "round(Lateral.Accel,1)")#
TEMPTRAIN <- menuTransform(data = TEMPTRAIN, variable.name = "RA", expression =  "ifelse(Radar.Altitude>100,NA,round(Radar.Altitude,0))")
TEMPTRAIN <- menuTransform(data = TEMPTRAIN, variable.name = "NR", expression = "round(Nr,1)")

TEMPTEST <- menuTransform(data = TEMPTEST, variable.name = "VhFr", expression = "round(Airspeed.Vh.Fraction,1)")
TEMPTEST <- menuTransform(data = TEMPTEST, variable.name = "Tq", expression = "(Torque.1+Torque.2)/2")
TEMPTEST <- menuTransform(data = TEMPTEST, variable.name = "Roll", expression = "ifelse(Roll.Attitude>=-4&Roll.Attitude <=-2,0,round(Roll.Attitude,0))")
TEMPTEST <- menuTransform(data = TEMPTEST, variable.name = "AltR", expression = "ifelse(Altitude.Rate>=-300&Altitude.Rate <=300,0,round(Altitude.Rate,0))")
TEMPTEST <- menuTransform(data = TEMPTEST, variable.name = "Pitch", expression = "ifelse(Pitch.Attitude>=2&Pitch.Attitude <=5,0,round(Pitch.Attitude,0))")
TEMPTEST <- menuTransform(data = TEMPTEST, variable.name = "Yaw", expression = "ifelse(Yawrate>=-2&Yawrate<=2,0,round(Yawrate))")
TEMPTEST <- menuTransform(data = TEMPTEST, variable.name = "Vert", expression = "round(Vertical.Accel,1)")
TEMPTEST <- menuTransform(data = TEMPTEST, variable.name = "Lat", expression = "round(Lateral.Accel,1)")#
TEMPTEST <- menuTransform(data = TEMPTEST, variable.name = "RA", expression =  "ifelse(Radar.Altitude>100,NA,round(Radar.Altitude,0))")
TEMPTEST <- menuTransform(data = TEMPTEST, variable.name = "NR", expression = "round(Nr,1)")
```

```
#########################################################################################################
# Subsetting for fast and slow regimes                                                                  #
#########################################################################################################

menuSubset(data =TEMPTRAIN, subset.expression = "KCAS < 44.72 ", subset.columns = "<ALL>", result.type = "Data Set", subset.col.name = "Subset", save.name = "IN.THE.AIR.SLOW", show.p = SHOW.ON.SCREEN);
menuSubset(data =TEMPTEST, subset.expression = "KCAS < 44.72 ", subset.columns = "<ALL>", result.type = "Data Set", subset.col.name = "Subset", save.name = "IN.THE.AIR.SLOW.TEST", show.p = SHOW.ON.SCREEN);

guiExportData(FileName = "C:\\IN.THE.AIR.SLOW.csv", FileTypeDesc = "ASCII file - comma delimited (csv)", SourceDataFrame = "IN.THE.AIR.SLOW", ColNames = T, RowNames = F, Quotes = T, ASCIIDelimiter = ",", KeepOrDropList = "<ALL>", KeepOrDrop = "Keep selected", Rows = "<ALL>", ASCIIDateOutFormat = "M/d/yyyy", ASCIITimeOutFormat = "h:mm:ss tt", ASCIIDecimalPoint = "Period (.)", ASCIIThousandsSeparator = "None")
guiImportData(FileName = "C:\\IN.THE.AIR.SLOW.csv", FileTypes = "ASCII file - comma delimited (csv)", TargetDataFrame = "IN.THE.AIR.SLOW.2", ImportAsBigData = F, TargetStartCol = "<END>", TargetInsertOverwrite = "Create new data set", NameRowAuto = "Auto", NameColAuto = "Auto", StartCol = 1, EndCol = "<END>", StartRow = 1, EndRow = "<END>", PageNumberAuto = "Auto", StringsAsFactors = T, SortFactorLevels = T, LabelsAsNumbers = F, CenturyCutoffYear = 1930, ASCIIDelimiters = "Comma (,)", KeepOrDropList = "&|", SeparateDelimiters = T, ASCIIDateInFormat = "M/d/yyyy", ASCIITimeInFormat = "h:mm:ss tt", ASCIIDecimalPoint = "Period (.)", ASCIIThousandsSeparator = "None", MissingValueString = "NA", LookMaxLinesString = "256", MaxLineWidth = 32768, SubsetNone = T, SubsetRandomSample = F, SubsetRandomSampleValue = 10, SubsetSampleNthRow = F, SubsetSampleNthRowValue = 10, SubsetKeepExpression = F)
guiExportData(FileName = "C:\\IN.THE.AIR.SLOW.TEST.csv", FileTypeDesc = "ASCII file - comma delimited (csv)", SourceDataFrame = "IN.THE.AIR.SLOW.TEST", ColNames = T, RowNames = F, Quotes = T, ASCIIDelimiter = ",", KeepOrDropList = "<ALL>", KeepOrDrop = "Keep selected", Rows = "<ALL>", ASCIIDateOutFormat = "M/d/yyyy", ASCIITimeOutFormat = "h:mm:ss tt", ASCIIDecimalPoint = "Period (.)", ASCIIThousandsSeparator = "None")
guiImportData(FileName = "C:\\IN.THE.AIR.SLOW.TEST.csv", FileTypes = "ASCII file - comma delimited (csv)", TargetDataFrame = "IN.THE.AIR.SLOW.TEST.2", ImportAsBigData = F, TargetStartCol = "<END>", TargetInsertOverwrite = "Create new data set", NameRowAuto = "Auto", NameColAuto = "Auto", StartCol = 1, EndCol = "<END>", StartRow = 1, EndRow = "<END>", PageNumberAuto = "Auto", StringsAsFactors = T, SortFactorLevels = T, LabelsAsNumbers = F, CenturyCutoffYear = 1930, ASCIIDelimiters = "Comma (,)", KeepOrDropList = "&|", SeparateDelimiters = T, ASCIIDateInFormat = "M/d/yyyy", ASCIITimeInFormat = "h:mm:ss tt", ASCIIDecimalPoint = "Period (.)", ASCIIThousandsSeparator = "None", MissingValueString = "NA", LookMaxLinesString = "256", MaxLineWidth = 32768, SubsetNone = T, SubsetRandomSample = F, SubsetRandomSampleValue = 10, SubsetSampleNthRow = F, SubsetSampleNthRowValue = 10, SubsetKeepExpression = F)
```

```
#########################################################################################################
# Load library rpart
#########################################################################################################
out <- try (library (rpart, lib.loc="r:/common/whitaker") )  # (Buttrey,2005)
 if (class (out) == "Error")  library (rpart, lib.loc="C:/Documents and Settings/Murat/My Documents/dersler/this/datamining_R_whitaker")
############################
# IN THE AIR SLOW TREE    #
############################
IN.THE.AIR.TREE.SLOW <- rpart(formula = Regime ~VhFr +AltR+ CONTROL.REVERSAL.ID+ Landing.Flag+ Lat+Pitch +Roll+Tq +Yaw+NR+RA, data = IN.THE.AIR.SLOW.2,
cp =0.001)
## Prune with 1-SE Rule; find the cp where xerror < (best xerror + best xerror's corresponding xstd) #(Ripley,B.,2004 June 7)
prune.1se <- function(intree) {
# Autoprune with 1 SE rule (Breiman 1984)
# Written by V. Bahn
cp.table <- intree$cptable
min.error <- min(cp.table[,4])
one.se <- cp.table[cp.table[,4] == min.error, 5]
cp.range <- cp.table[cp.table[,4] < min.error + one.se, 1]
cp.1se <- max(cp.range)
temp.tree <- prune(intree, cp = cp.1se)
temp.tree
}
IN.THE.AIR.TREE.SLOW_prune.1se(IN.THE.AIR.TREE.SLOW)
graphsheet();plotcp(IN.THE.AIR.TREE.SLOW)
title("IN THE AIR AND SLOW ")
graphsheet();plot(IN.THE.AIR.TREE.SLOW,branch=.4,compress=T)
title(" IN THE AIR AND SLOW ")
text(IN.THE.AIR.TREE.SLOW)
```

```
# Takes out the predicted vector and turn it into a table and data.frame.
predAS_data.frame(table(IN.THE.AIR.SLOW.TEST.2$Regime,predict(IN.THE.AIR.TREE.SLOW,IN.THE.AIR.SLOW.TEST.2,type="vector")))
predtestAS_data.frame(table(IN.THE.AIR.SLOW.TEST.2$Regime,IN.THE.AIR.SLOW.TEST.2$Regime))

#########################################################################################################
#Penalty or loss matrix is constructed using the predicted values.Wherever there is misclassification,depending          #
#on the distance between regimes (This is also the answer of are they in the same family? or How bad is the misclassification?#                                  #
#########################################################################################################
temp_predAS
temp_as.matrix(temp)
# Decides the penalty close neighbors small penalty distant neighbors big penalty

decidePenalty_function(i,j){
  delta_abs(i-j)
 if(delta <= 2 ) x_1.1 else if(delta == 3) x_1.5 else if(delta == 4) x_2 else x_3
 x
}

#Transforms the diagonals to zero and assigns a bigger penalty if the number is bigger than a threshold value using on the distance between the actual and the predicted regime
formTheMatrix_function(x){

 for(i in 1:15){
   for(j in 1:15){
     if (i==j) x[i,j]_0 else   if( x[i,j]>7 ) x[i,j]_decidePenalty(i,j) else x[i,j]_1
   }
 }
x
}

ASloss_formTheMatrix(temp)
```

```
#########################################################################################################################
# Here using the cost matrix,a new tree is constructed. Finding out the bad misclassifications,it is hoped that we wont see those bad ones again.  #
# Only acceptable misclassifications are good;and it means that misclass. is close to the actual regime.                                          #
#########################################################################################################################
IN.THE.AIR.TREE.SLOW <- rpart(formula = Regime ~VhFr +AltR+CONTROL.REVERSAL.ID+Landing.Flag+Lat+Pitch+Roll+Tq+Yaw+Nr+RA,parms=list(loss=ASloss) , data =
IN.THE.AIR.SLOW.2,  cp =0.001)
#1-SE rule
IN.THE.AIR.TREE.SLOW_prune.1se(IN.THE.AIR.TREE.SLOW)
graphsheet();plotcp(IN.THE.AIR.TREE.SLOW)
title("IN THE AIR SLOW with penalties ")
graphsheet();plot(IN.THE.AIR.TREE.SLOW,branch=.4,compress=T)
title("   IF(Weight on Wheels = 0 AND SLOW with penalties ")
text(IN.THE.AIR.TREE.SLOW)

#predicted values
predASL_data.frame(table(IN.THE.AIR.SLOW.TEST.2$Regime,predict(IN.THE.AIR.TREE.SLOW,IN.THE.AIR.SLOW.TEST.2,type="vector")))

#test values (what should have been observed for fitted values?)
predtestAS_data.frame(table(IN.THE.AIR.SLOW.TEST.2$Regime,IN.THE.AIR.SLOW.TEST.2$Regime))

#plot a simple tree
AS2_snip.rpart(IN.THE.AIR.TREE.SLOW,toss=50:2000);graphsheet();plot(AS2,branch=0);text(AS2);title("SIMPLIFIED ~IN THE AIR AND MOVING SLOW~ TREE")


#Correct classification rate
sum(diag( predASL))/sum(diag( predtestAS))
```

The Differences in the Penalty function for Different Models (Replace the bold lines with the ones given below.)

```
IN.THE.AIR.TREE.FAST <- rpart(formula = Regime ~VhFr + AltR+ CONTROL.REVERSAL.ID+ Landing.Flag+Lat+Pitch+Roll+Tq+ Yaw+NR+RA, data = IN.THE.AIR.FAST.2, cp
=0.001)

decidePenalty_function(i,j){
   delta_abs(i-j)
   if(delta <= 2 ) x_1.1 else if(delta == 3) x_1.5 else if(delta == 4) x_2 else x_3
   x
}
```

The Differences in scripts for "ON THE GROUND" Tree.(Replace the bold lines with the ones given below)

```
ON.THE.GROUND.TREE  <- rpart(formula = Regime ~Lat+Tq+Yaw+NR+RA, data = ON.THE.GROUND.2,  cp =0.001)

decidePenalty_function(i,j){
   delta_abs(i-j)
  if(delta = 1 ) x_1 else if(delta == 2) x_2 else if(delta == 3) x_3
  x
}
```

LIST OF REFERENCES

Angshuman, S. (n.d.).Tutorial on Neural Network Based Modeling, Retrieved October 10, 2006 from http://www.geocities.com/adotsaha/NNinExcel.html

Buttrey, S.E. (2005).The try() Function. Retrieved September 25, 2006 from http://web.nps.navy.mil/~buttrey/S/tipsandtricks.html

Devore, J.L.(2004).Probability and Statistics, Sixth Edititon, Chapter 7, Thompson Learning

Documentation and Reference Notes for SPSS Clementine 10.0 Software
Duda, Hart, &Stork.(2002).Pattern Classification(2nd ed)., New York: John Wiley & Sons Inc.

Garson D., (1998).Topics in Multivariate Analysis, Retrieved October 10, 2006 from http://www2.chass.ncsu.edu/garson/pa765/logistic.htm

Goodrich Corporation. (2002a). Configuration Requirements Specification for Army UH-60L IMDS Usage. Vermont: Cole

Goodrich Corporation.(2001).USCG HIDS Report Rev2.Vermont:Chin.

Goodrich Corporation.(2002b).Configuration Detailed Design for Army UH-60L IMDS Usage. Vermont: Cole

Goodrich Corporation. (n.d.). Usage Regime Exampled. Vermont: Bechhoefer

Kdnuggets Editor.(2006,March 11).C5.0 Dec-Tree Algorithm Explaination. [Msg] Message posted to http://www.kdnuggets.com/phpBB/viewtopic.php?p=189&,

Lanzi,P.L.(2003).Classification:Decision Trees.
http://www.elet.polimi.it/upload/lanzi/taadm/gray/Unit%2003%20-%20Classification%20Trees.pdf Retrieved September 24, 2006

Operator's Manual for UH-60A Helicopter.(1996).Headquarters Department Of The Army. Washington, D.C

Ripley, B.(2004,June 7).Re: error in rpart pruning [Msg 00048].Message posted to http://www.biostat.wustl.edu/archives/html/s-news/2004-06/msg00048.html

SPSS Whitepapers. (1999).SPSS AnswerTree Whitepaper. Retrieved September 22, 2006 from http://www.spss.com/downloads/Papers.cfm)

StatSoft Inc. (n.d.a) Electronic statistics textbook. Retrieved October 5, 2006, from http://www.statsoft.com/textbook/stchaid.html#overview
StatSoft Inc. (n.d.b) Electronic statistics textbook.Retrieved December 1, 2006, from http://www.statsoft.com/textbook/stcart.html#overview

Teal,R.S.,Evernham J.T., Larchuk, T.J.,Miller, D.G.,Marquith, D.E., White, F.(1997).Proceedings from American Helicopter Society 53rd Annual Forum(Vols.2), Virginia Beach

Therneau, T.M. & Atkinson.(2000).An Introduction to Recursive Partitioning Using the RPART Routines Mayo Foundation. Retrieved October 3, 2006 from http://cancercenter.mayo.edu/mayo/research/biostat/upload/rpartmini.pdf

Webb, A.R.(2002).Statistical Pattern Recognition(2nd ed.).West Sussex: John Wiley & Sons Ltd.

Whitaker, Lyn.(2006) Summer 2006 Data mining Class/Lab Notes. Retrieved November 1,2006 from NPS Intranet R:\\Whitaker

Wikipedia: The free encyclopedia. (2006, February 13). FL: Wikipedia Foundation, Inc. Retrieved September 25, 2006, from http://www.wikipedia.org

Williams, G. (2004).Predicting Fraud: Underrepresented Classes. Retrieved September 24, 2006 from
http://www.togaware.com/datamining/survivor/Predicting_Fraud.html .

INITIAL DISTRIBUTION LIST

1. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

2. Professor Samuel E. Buttrey
   Department of Operations Research
   Naval Postgraduate School
   Monterey, CA

3. Professor Lyn R. Whitaker
   Department of Operations Research
   Naval Postgraduate School
   Monterey, CA